VISIT…

LANZAROTE
Caliente.COM

计算机科学丛书

Discrete Mathematics and Its Applications

离散数学及其应用

（英文版 · 第4版）

（美）Kenneth H. Rosen 著

机械工业出版社
China Machine Press
McGraw-Hill

计算机科学丛书

离散数学及其应用

（英文版 第4版）

Discrete Mathematics and Its Applications

（美） Kenneth H. Rosen 著

机械工业出版社
China Machine Press
WCB
McGraw-Hill

Kenneth H. Rosen: Discrete Mathematics and Its Applications, Fourth Edition

Copyright © 1998 by The McGraw-Hill Companies, Inc. All rights reserved, Jointly published by China Machine Press/McGraw-Hill. This edition may be sold in the People's Republic of China only. This book cannot be re-exported and is not for sale outside the People's Republic of China.

RISBN: 007-1168818

本书英文影印版由McGraw-Hill 公司授权机械工业出版社在中国大陆境内独家出版发行，未经出版者书面许可，不得以任何方式抄袭、复制或节录本书中的任何部分。

版权所有，侵权必究。

本书版权登记号：图字：01-1999-1418

图书在版编目(CIP)数据

离散数学及其应用：第4版：英文/（美）罗森（Rosen, K, H.）著. －影印版. －北京：机械工业出版社，1999.6
（计算机科学丛书）
ISBN 7-111-07254-5

Ⅰ.离… Ⅱ.罗… Ⅲ.离散数学－英文 Ⅳ.O158

中国版本图书馆CIP数据核字（1999）第16772号

出 版 人：马九荣（北京市西城区百万庄大街22号 邮政编码 100037）
北京市密云县印刷厂印刷 · 新华书店北京发行所发行
1999年6月第1版第1次印刷
787mm × 1092mm 1/16 · 45.25印张
印数：0 001- 5 000册
定价：59.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

ABOUT THE AUTHOR

Kenneth H. Rosen is a Distinguished Member of the Technical Staff in the New Concepts Area of AT&T Laboratories in Holmdel, New Jersey.

Dr. Rosen received his B.S. in Mathematics from the University of Michigan, Ann Arbor (1972), and his Ph.D. in Mathematics from M.I.T. (1976), where he wrote his thesis in the area of number theory under the direction of Harold Stark. Before joining Bell Laboratories in 1982, he held positions at the University of Colorado, Boulder; the Ohio State University, Columbus; and the University of Maine, Orono, where he was an associate professor of mathematics. While working at AT&T Labs, Ken has taught in the evening program in computer science at Monmouth University, teaching courses in discrete mathematics, coding theory, and data security.

Dr. Rosen has published numerous articles in professional journals in the areas of number theory and mathematical modeling. He is the author of the textbooks *Elementary Number Theory and Its Applications*, currently in its third edition, published by Addison-Wesley, and *Discrete Mathematics and Its Applications*, in its fourth edition, published by McGraw-Hill. Both books have been used extensively at hundreds of universities. He is coauthor of *UNIX System V Release 4: An Introduction*, which has sold more than 100,000 copies and has been translated into Spanish and German, and *Best UNIX Tips Ever*, translated into Chinese, both published by Osborne McGraw-Hill. Ken is also the editor of the *Handbook of Discrete Mathematics*, a new publication to be published in 1999 by CRC Press, and he is the editor of the CRC series of books in discrete mathematics. Ken is also interested in integrating mathematical software into the educational and professional environments and is working on projects with Waterloo MAPLE software in both these areas.

At Bell Laboratories and now AT&T Laboratories, Dr. Rosen has worked on a wide range of projects, including operations research studies and product line planning for computers and data communications equipment. He has helped plan AT&T's future products and services in the area of multimedia, including video communications, speech recognition, and image networking. He has evaluated new technology for use by AT&T. He has also invented many new services and holds or has submitted many patents. One of his more interesting projects involved helping evaluate technology for the AT&T attraction at EPCOT Center.

PREFACE

In writing this book, I was guided by my long-standing experience and interest in teaching discrete mathematics. For the student, my purpose was to present material in a precise, readable manner, with the concepts and techniques of discrete mathematics clearly presented and demonstrated. My goal was to show the relevance and practicality of discrete mathematics to students, who are often skeptical. I wanted to give students studying computer science all the mathematical foundations they need for their future studies; I wanted to give mathematics students an understanding of important mathematical concepts together with a sense of why these concepts are important for applications. And I wanted to accomplish these goals without watering down the material.

For the instructor, my purpose was to design a flexible, comprehensive teaching tool using proven pedagogical techniques in mathematics. I wanted to provide instructors with a package of materials that they could use to teach discrete mathematics effectively and efficiently in the most appropriate manner for their particular set of students. I hope that I have achieved these goals.

I have been extremely gratified by the tremendous success of this text. The many improvements in the fourth edition have been made possible by the feedback and suggestions of a large number of instructors and students at many of the more than 400 schools where this book has been successfully used. There are many enhancements in this edition. The ancillary package has been enriched, and a companion Web site provides helpful material, making it easier for students and instructors to achieve their goals.

This text is designed for a one- or two-term introductory discrete mathematics course to be taken by students in a wide variety of majors, including mathematics, computer science, and engineering. College algebra is the only explicit prerequisite.

Goals of a Discrete Mathematics Course

A discrete mathematics course has more than one purpose. Students should learn a particular set of mathematical facts and how to apply them; more importantly, such a course should teach students how to think mathematically. To achieve these goals, this text stresses mathematical reasoning and the different ways problems are solved. Five important themes are interwoven in this text: mathematical reasoning, combinatorial analysis, discrete structures, algorithmic thinking, and applications and modeling. A successful discrete mathematics course should carefully blend and balance all five themes.

1. *Mathematical Reasoning:* Students must understand mathematical reasoning in order to read, comprehend, and construct mathematical arguments. This text starts with a discussion of mathematical logic, which serves as the foundation for the

subsequent discussions of methods of proof. The technique of mathematical induction is stressed through many different types of examples of such proofs and a careful explanation of why mathematical induction is a valid proof technique.

2. *Combinatorial Analysis:* An important problem-solving skill is the ability to count or enumerate objects. The discussion of enumeration in this book begins with the basic techniques of counting. The stress is on performing combinatorial analysis to solve counting problems, not on applying formulae.
3. *Discrete Structures:* A course in discrete mathematics should teach students how to work with discrete structures, which are the abstract mathematical structures used to represent discrete objects and relationships between these objects. These discrete structures include sets, permutations, relations, graphs, trees, and finite-state machines.
4. *Algorithmic Thinking:* Certain classes of problems are solved by the specification of an algorithm. After an algorithm has been described, a computer program can be constructed implementing it. The mathematical portions of this activity, which include the specification of the algorithm, the verification that it works properly, and the analysis of the computer memory and time required to perform it, are all covered in this text. Algorithms are described using both English and an easily understood form of pseudocode.
5. *Applications and Modeling:* Discrete mathematics has applications to almost every conceivable area of study. There are many applications to computer science and data networking in this text, as well as applications to such diverse areas as chemistry, botany, zoology, linguistics, geography, business, and the Internet. These applications are natural and important uses of discrete mathematics and are not contrived. Modeling with discrete mathematics is an extremely important problem-solving skill, which students have the opportunity to develop by constructing their own models in some of the exercises in the book.

Why a Fourth Edition?

The third edition of this book has been used successfully at over 400 schools in the United States, at dozens of Canadian universities, and at universities in Europe, Asia, and Oceania. Many students and professors like the third edition as it is. Why then, do we need a fourth edition? This is a valid question deserving a careful answer.

First, although the third edition has been extremely effective, many instructors have asked for specific improvements. Many have wanted changes to the text, additional or clarified examples, more exercises of a certain type, or new topics covered. In this new edition I have improved the book by taking into account the numerous suggestions I have received. The changes I have made at the request of users make this a better text.

Second, discrete mathematics is an active subject. There are many new discoveries made every year, and some of these can be reflected in a text. So, I have included discoveries made after the publication of the third edition. (Subsequent discoveries will be included in later printings of this edition whenever possible and noted on the companion Web site.)

Third, since the publication of the third edition, the Internet has become extraordinarily important and useful. In this edition you will find examples and exercises relating applications of discrete mathematics to the structure of the Internet itself. And with this edition there is an extensive Web site that supplements the text in meaningful ways, offering additional material for students and instructors and providing a gateway for learning more about discrete mathematics by providing links to relevant sites on the Web. However, since many people will choose not to use the Web in conjunction with this course, the text includes icons indicating the inclusion of Web links in the annotated Web Guide on the Web site for this book.

The following list highlights some of the changes in this edition that make the book more effective.

NEW TOPIC COVERAGE

- Big-Omega and big-Theta notation are now covered, in addition to big-O notation.
- New topics in probability theory include the variance of a random variable and Chebyshev's inequality. Also, the Monty Hall three-door problem is now discussed in the text.
- The halting problem is now treated, including a proof that it is unsolvable.
- The traveling salesman problem is discussed.

EXPANDED TOPIC COVERAGE

- Additional material on mathematical logic and mathematical reasoning has been added. New examples show how to translate between quantified statements and English. The discussion of rules of inference has been enhanced. In particular, rules of inference for quantified statements are now explicitly covered, and examples illustrating how rules of inference are used have been added.
- Coverage of the floor and ceiling functions has been enhanced.
- Generating functions are now treated in a separate section in the main body of the text, expanding the coverage previously found in the appendix. The focus of this section is to show how generating functions can be used to solve counting problems, solve recurrence relations, and prove combinatorial identities.
- Nonhomogeneous linear recurrence relations with constant coefficients are now discussed in the text, rather than in an exercise set.
- The topic of integer sequences has extended coverage; examples and exercises involving identifying possible formulas for the terms of a sequence from its initial terms have been added.
- New biographies have been added, including those for Peirce, Chebyshev, Knuth, Hardy, Ramanujan, Tukey, Sloane, and Mersenne.

UP-TO-DATE, MODERN EXAMPLES

- Examples have been added at some key points in the text to help explain important concepts that have proved troublesome to students and to make the book more interesting.
- Examples and exercises illustrating the application of discrete mathematics to the protocols and network architecture of the Internet have been added. These additions include counting problems involving Internet addresses and Internet Protocol packets; the topic of Boolean searching, used by Internet search engines; and an example about how spanning trees are used in IP multicasting have been added.

- Material has been added to the text which demonstrates that discrete mathematics is an active subject with many open questions and with new discoveries. For example, Mersenne primes are now covered, including the discoveries of new primes in 1997 and 1998; the range for which the Goldbach conjecture has been verified is discussed; and the variation of the Tower of Hanoi puzzle with four pegs is described.

EXPANDED EXERCISE SETS

- More than 500 new exercises have been added, including both routine and challenging ones, as requested by instructors who used the third edition, as well as exercises based on logical and mathematical puzzles. New blocks of exercises develop key concepts in a series of steps. New exercises ensure that there are both odd- and even-numbered exercises of important exercise types. There are also more exercises that depend on the previous study of calculus; these are explicitly noted as usual and can be easily avoided if so desired.

WEB SUPPORT

- A Web site has been developed to supplement the text for both students and instructors. This Web site contains a wide range of features (see page xix), including an annotated Web Guide to relevant sites on the Internet, that is keyed to the text. This guide will be kept current and updated regularly during the life of this edition.

web

- An icon has been placed at points in the text whenever the Web Guide includes annotated links to Web sites pertinent to the material under discussion. (More than 200 different links are in the guide.) These sites include additional information about concepts and applications, biographies, the latest discoveries, downloadable source code, interactive applets, animated algorithms, and other interesting material.

Special Features

ACCESSIBILITY This text has proven to be easily read and understood by beginning students. There are no mathematical prerequisites beyond college algebra for almost all of this text. The few places in the book where calculus is referred to are explicitly noted. Most students should easily understand the pseudocode used in the text to express algorithms, regardless of whether they have formally studied programming languages. There is no formal computer science prerequisite.

Each chapter begins at an easily understood and accessible level. Once basic mathematical concepts have been carefully developed, more difficult material and applications to other areas of study are presented.

FLEXIBILITY This text has been carefully designed for flexible use. The dependence of chapters on previous material has been minimized. Each chapter is divided into sections of approximately the same length, and each section is divided into subsections that form natural blocks of material for teaching. Instructors can easily pace their lectures using these blocks.

WRITING STYLE The writing style in this book is direct and pragmatic. Precise mathematical language is used without excessive formalism and abstraction. Notation is introduced and used when appropriate. Care has been taken to balance the mix of notation and words in mathematical statements.

EXTENSIVE CLASSROOM USE This book has been used at over 400 schools, and more than 325 have used it more than once. The feedback from instructors and students at many of the schools has helped make the fourth edition an even more successful teaching tool than previous editions.

MATHEMATICAL RIGOR AND PRECISION All definitions and theorems in this text are stated extremely carefully so that students will appreciate the precision of language and rigor needed in mathematics. Proofs are motivated and developed slowly; their steps are all carefully justified. Recursive definitions are explained and used extensively.

FIGURES AND TABLES This text contains more than 550 figures. The figures are designed to illustrate key concepts and steps of proofs. Color has been carefully used in figures to illustrate important points. Whenever possible, tables have been used to summarize key points and illuminate quantitative relationships.

WORKED EXAMPLES Over 650 examples are used to illustrate concepts, relate different topics, and introduce applications. In the examples, a question is first posed, then its solution is presented with the appropriate amount of detail.

APPLICATIONS The applications included in this text demonstrate the utility of discrete mathematics in the solution of real-world problems. This text includes applications to a wide variety of areas, including computer science, data networking, psychology, chemistry, engineering, linguistics, biology, business, and the Internet.

ALGORITHMS Results in discrete mathematics are often expressed in terms of algorithms; hence, key algorithms are introduced in each chapter of the book. These algorithms are expressed in words and in an easily understood form of structured pseudocode, which is described and specified in Appendix A.2. The computational complexity of the algorithms in the text is also analyzed at an elementary level.

HISTORICAL INFORMATION The background of many topics is succinctly described in the text. Brief biographies of more than 55 mathematicians and computer scientists are included as footnotes. These biographies include information about the lives, careers, and accomplishments of these important contributors to discrete mathematics. In addition, numerous historical footnotes are included that supplement the historical information in the main body of the text.

KEY TERMS AND RESULTS A list of key terms and results follows each chapter. The key terms include only the most important that students should learn, not every term defined in the chapter.

EXERCISES There are over 3000 exercises in the text. There are many different types of questions posed. There is an ample supply of straightforward exercises that develop basic skills, a large number of intermediate exercises, and many challenging exercises. Exercises are stated clearly and unambiguously, and all are carefully graded for level of difficulty. Exercise sets contain special discussions, with exercises, that develop new concepts not covered in the text, permitting students to discover new ideas through their own work.

Exercises that are somewhat more difficult than average are marked with a single star; those that are much more challenging are marked with two stars. Exercises whose solutions require calculus are explicitly noted. Exercises that develop results used in the text are clearly identified with the symbol ☞. Answers or outlined solutions to all odd-numbered exercises are provided at the back of the text. The solutions include proofs in which most of the steps are clearly spelled out.

REVIEW QUESTIONS A set of review questions is provided at the end of each chapter. These questions are designed to help students focus their study on the most important concepts and techniques of that chapter. To answer these questions students need to write long answers, rather than just perform calculations or give short replies.

SUPPLEMENTARY EXERCISE SETS Each chapter is followed by a rich and varied set of supplementary exercises. These exercises are generally more difficult than those in the exercise sets following the sections. The supplementary exercises reinforce the concepts of the chapter and integrate different topics more effectively.

COMPUTER PROJECTS Each chapter is followed by a set of computer projects. The approximately 150 computer projects tie together what students may have learned in computing and in discrete mathematics. Computer projects that are more difficult than average, from both a mathematical and a programming point of view, are marked with a star, and those that are extremely challenging are marked with two stars.

COMPUTATIONS AND EXPLORATIONS A set of computations and explorations is included at the conclusion of each chapter. These exercises (approximately 100 in total) are designed to be completed using existing software tools, such as programs that students or instructors have written or mathematical computation packages such as MAPLE or Mathematica. Many of these exercises give students the opportunity to uncover new facts and ideas through computation. (Some of these exercises are discussed in the companion volume, *Exploring Discrete Mathematics with MAPLE.*)

WRITING PROJECTS Each chapter is followed by a set of writing projects. To do these projects students need to consult the mathematical literature. Some of these projects are historical in nature and may involve looking up original sources. Others are designed to serve as gateways to new topics and ideas. All are designed to expose students to ideas not covered in depth in the text. These projects tie together mathematical concepts and the writing process and help expose students to possible areas for future study. (Suggested references for these projects can be found in the *Student Solutions Guide.*)

APPENDIXES There are two appendixes to the text. The first covers exponential and logarithmic functions, reviewing some basic material used heavily in the course; the second specifies the pseudocode used to describe algorithms in this text.

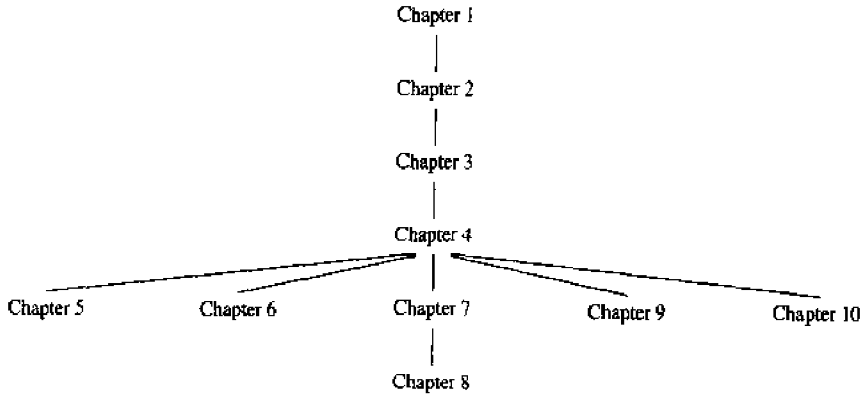
SUGGESTED READING A list of suggested readings for each chapter is provided in a section at the end of the text. These suggested readings include books at or below the level of this text, more difficult books, expository articles, and articles in which discoveries in discrete mathematics were originally published.

How To Use This Book

This text has been carefully written and constructed to support discrete mathematics courses at several levels and with differing foci. The following table identifies the core and optional sections. An introductory one-term course in discrete mathematics at the sophomore level can be based on the core sections of the text, with other sections covered at the discretion of the instructor. A two-term introductory course could include all the optional mathematics sections in addition to the core sections. A course with a strong computer science emphasis can be taught by covering some or all of the optional computer science sections.

Chapter	*Core Sections*	*Optional Computer Science Sections*	*Optional Mathematics Sections*
1	1.1–1.8 (as needed)		
2	2.1–2.3, 2.6 (as needed)	2.4	2.5
3	3.1–3.3	3.4, 3.5	
4	4.1–4.4	4.7	4.5, 4.6
5	5.1, 5.5	5.3	5.2, 5.4, 5.6
6	6.1, 6.3, 6.5	6.2	6.4, 6.6
7	7.1–7.5		7.6–7.8
8	8.1	8.2–8.4	8.5, 8.6
9		9.1–9.4	
10		10.1–10.5	

Instructors using this book can adjust the level of difficulty of their course by either choosing to cover or to omit the more challenging examples at the end of sections, as well as the more challenging exercises. The dependence of chapters on earlier chapters is shown in the following chart.



Ancillaries

STUDENT SOLUTIONS GUIDE This student manual, available separately, contains *full* solutions to all the odd-numbered problems in the exercise sets. These solutions explain why a particular method is used and why it works. For some exercises, one or two other possible approaches are described to show that a problem can be solved in several different ways. Suggested references for the writing projects found at the end of each chapter are also included in this volume. The guide contains a guide to writing proofs and a list of common mistakes students make in discrete mathematics. It also includes sample tests and a sample crib sheet for each chapter, both designed to help students prepare for exams. Students find this guide extremely useful.

INSTRUCTOR'S RESOURCE GUIDE This manual contains full solutions to even-numbered exercises in the text. It also provides suggestions on how to teach the material in each chapter of the book, including the points to stress in each section and how to put the material into perspective. Furthermore, the manual contains a test bank of sample examination questions for each chapter, including some sample tests as well as the solutions to the sample questions. Finally, sample syllabi are presented.

APPLICATIONS OF DISCRETE MATHEMATICS This ancillary is a separate text that can be used either in conjunction with the text or independently. It contains more than 20 chapters (each with its own set of exercises) written by instructors who have used the text. Following a common format similar to that of the text, the chapters in this book can be used as a text for a separate course, for a student seminar, or for a student doing independent study. Subsequent editions of this ancillary are planned that will broaden the range of applications covered. Instructors are invited to submit additional applications for possible inclusion in later versions.

TEST BANK An extensive test bank of more than 1300 questions is available for use on either Windows or Macintosh systems. Instructors can use this software to create their own tests by selecting questions of their choice or by random selection. Instructors can add their own headings and instructions, print scrambled versions of the same test, and edit the existing questions or add their own. A printed version of this test bank, including the questions and their answers, is included in the Instructor's Resource Guide.

EXPLORING DISCRETE MATHEMATICS AND ITS APPLICATIONS WITH MAPLE This ancillary is a separate book designed to help students use the MAPLE computer algebra system to do a wide range of computations in discrete mathematics. For each chapter of this text, this new ancillary includes the following: a description of relevant MAPLE functions and how they are used, MAPLE programs that carry out relevant computations, suggestions and examples showing how MAPLE can be used for the computations and explorations at the end of each chapter, and exercises that can be worked using MAPLE.

Acknowledgments

I would like to thank the many instructors and students at many different schools who have used this book and provided me with their valuable feedback and helpful suggestions. Their input has made this a much better book than it would have been otherwise. I especially want to thank Jerrold Grossman and John Michaels for their technical reviews of the fourth edition and their "eagle eyes," which have helped ensure the accuracy of this book.

I thank the many, many reviewers of the first, second, third, and fourth editions. These reviewers have provided much helpful criticism and encouragement to me. I hope this edition lives up to their high expectations.

Eric Allender,
Rutgers University
Stephen Andrilli,
La Salle University
Jack R. Barone,
Baruch College
Alfred E. Borm,
Southwest Texas State University
Ken W. Bosworth,
University of Maryland
Lois Brady,
California Polytechnic State University, San Luis Obispo
Russell Campbell,
University of Northern Iowa
Kevin Carolan,
Marist College
Tim Carroll,
Bloomsburg University
Kit C. Chan,
Bowling Green State University
Allan C. Cochran,
University of Arkansas
Peter Collinge,
Monroe Community College
Ron Davis,
Millersville University
Nachum Dershowitz,
University of Illinois, Urbana-Champaign
Thomas Dowling,
Ohio State University
Patrick C. Fischer,
Vanderbilt University
Jane Fritz,
University of New Brunswick
Ladnor Geissinger,
University of North Carolina
Paul Gormley,
Villanova University
Jerrold Grossman,
Oakland University
Laxmi N. Gupta,
Rochester Institute of Technology
Daniel Gusfield,
University of California at Davis
Xin He,
State University of New York at Buffalo
Arthur M. Hobbs,
Texas A&M University
Donald Hutchison,
Clackamas Community College
Kenneth Johnson,
North Dakota State University
David Jonah,
Wayne State University
W. Thomas Kiley,
George Mason University
Nancy Kinnersley,
University of Kansas
Gary Klatt,
University of Wisconsin
Nicholas Krier,
Colorado State University
Lawrence S. Kroll,
San Francisco State University
Robert Lavelle,
Iona College
Yi-Hsin Liu,
University of Nebraska at Omaha
George Luger,
University of New Mexico
David S. McAllister,
North Carolina State University
Robert McGuigan,
Westfield State College
Michael Maller,
Queens College
Ernie Manes,
University of Massachusetts
Francis Masat,
Glassboro State College

J. M. Metzger,
University of North Dakota
John G. Michaels,
SUNY Brockport
D. R. Morrison,
University of New Mexico
Ho Kuen Ng,
San Jose State University
Timothy S. Norfolk, Sr.,
University of Akron
Jeffrey Nunemacher,
Ohio Wesleyan University
Charles Parry,
Virginia Polytechnic Institute and State University
Thomas W. Parsons,
Hofstra University
Mary K. Prisco,
University of Wisconsin–Green Bay
Harold Reiter,
University of North Carolina
Adrian Riskin,
Northern Arizona State University
Amy L. Rocha,
San Jose State University
Janet Roll,
University of Findlay
Alyssa Sankey,
Slippery Rock University
Dinesh Sarvate,
College of Charleston
Steven R. Seidel,
Michigan Technological University
Douglas Shier,
Clemson University
Hunter S. Snevily,
University of Idaho
Daniel Somerville,
University of Massachusetts, Boston
Bharti Temkin,
Texas Tech University
Wallace Terwilligen,
Bowling Green State University
Philip D. Tiu,
Oglethorpe University
Lisa Townsley-Kulich,
Illinois Benedictine College
George Trapp,
West Virginia University
David S. Tucker,
Midwestern State University
Thomas Upson,
Rochester Institute of Technology
Roman Voronka,
New Jersey Institute of Technology
James Walker,
University of South Carolina
Anthony S. Wojcik,
Michigan State University

I would like to thank the staff of the McGraw-Hill Higher Education Group for their support of this project. In particular, thanks go to J. P. Lenney, Publisher, for his overall support; to Maggie Rogers, Sponsoring Editor, for her continued interest and enthusiasm; to Nina Kreiden, Developmental Editor, for her dedication and diligence; and to Amy Upgren, Editorial Assistant, for her valuable help. I would also like to thank the original editor, Wayne Yuhasz, whose insights and skills helped ensure the book's success, as as well as Jack Shira, Tom Casson, and Mike Morales, who helped plan and initiate work on the fourth edition.

To the staff involved in the production of this fourth edition, I offer my thanks to Rich DeVitto, Production Supervisor; Suzanne Montazer, Designer; Ronald Tigges, Web Developer; Louis Swaim, Supplements Coordinator; and Mary Kittell, Marketing Manager.

As always, I am grateful for the support given to me by my management at AT&T Laboratories. They have provided an environment that has helped me develop professionally and generously given me the resources needed to make this book a success.

Kenneth H. Rosen

THE COMPANION WEB SITE

An extensive companion Web site has been developed and will be maintained and improved on a continuing basis. The URL for this site is http://www.mhhe.com/rosen. Following this URL takes you to a page that provides access to five different sections of the Web site:

- About the Book
- Instructor Resources
- Student Resources
- Web Guide for Discrete Mathematics
- Supplementary Resources

Each section will be in place with the publication of this new edition, although additional material will be added later.

The *About the Book* section includes basic information about the textbook and its ancillaries. It also contains an errata list and an e-mail address for the submission of errata and suggestions.

The *Instructor Resources* section is a secure portion of the Web site. It contains valuable tools and resources to supplement both the text and the discrete mathematics teaching experience.

The *Student Resources* section contains helpful reference and supplemental material to enhance students' learning experience.

The *Web Guide for Discrete Mathematics* section includes annotated links to relevant Web sites anchored to the Web icons in the text. (Links are included wherever the icon is found.) The links in this guide can be used to access sites that provide biographies, additional material on topics covered in the text, information on the latest discoveries, animated algorithms, downloadable source code, and so on.

The *Supplementary Resources* section, intended for use by both students and instructors, includes supplementary educational material, organized by chapter. This material is designed to clarify and expand on material in the text.

TO THE STUDENT

What is discrete mathematics? Discrete mathematics is the part of mathematics devoted to the study of discrete objects. (Here *discrete* means consisting of distinct or unconnected elements.) The kind of problems solved using discrete mathematics include:

- How many ways are there to choose a valid password on a computer system?
- What is the probability of winning a lottery?
- Is there a link between two computers in a network?
- What is the shortest path between two cities using a transportation system?
- How can a list of integers be sorted so that the integers are in increasing order?
- How many steps are required to do such a sorting?
- How can a circuit that adds two integers be designed?
- How many valid Internet addresses are there?

You will learn the discrete structures and techniques needed to solve problems such as these.

More generally, discrete mathematics is used whenever objects are counted, when relationships between finite (or countable) sets are studied, and when processes involving a finite number of steps are analyzed. A key reason for the growth in the importance of discrete mathematics is that information is stored and manipulated by computing machines in a discrete fashion.

There are several important reasons for studying discrete mathematics. First, through this course you can develop your mathematical maturity, that is, your ability to understand and create mathematical arguments. You will not get very far in your studies in the mathematical sciences without these skills.

Second, discrete mathematics is the gateway to more advanced courses in all parts of the mathematical sciences. Discrete mathematics provides the mathematical foundations for many computer science courses, including data structures, algorithms, database theory, automata theory, formal languages, compiler theory, computer security, and operating systems. Students find these courses much more difficult when they have not had the appropriate mathematical foundations from discrete math. One student has sent me an electronic mail message to tell me that she used the contents of this book in every computer science course she took!

Math courses based on the material studied in discrete mathematics include logic, set theory, number theory, linear algebra, abstract algebra, combinatorics, graph theory, and probability theory (the discrete part of the subject).

Also, discrete mathematics contains the necessary mathematical background for solving problems in operations research (including many discrete optimization techniques), chemistry, engineering, biology, and so on. In the text, we will study applications to some of these areas.

I would like to offer some helpful advice to students about how best to learn discrete mathematics. You will learn the most by working exercises. I suggest you do as many as you possibly can, including both the exercises at the end of each section of the text and the supplementary exercises at the end of each chapter. Always try exercises yourself before consulting the answers at the end of the book or in the *Student Solutions Guide.* Only after you have put together a solution, or you find yourself at an impasse, should you look up the suggested solution. At that point you will find the discussions in the *Student Solutions Guide* most helpful. When doing exercises, remember that the more difficult ones are marked as described in the following table.

Key to the Exercises	
No marking	A routine exercise
*	A difficult exercise
**	An extremely challenging exercise
☞	An exercise containing a result used in the book
(Calculus required)	An exercise whose solution requires the use of limits or concepts from differential or integral calculus

Finally, I encourage you to explore discrete mathematics beyond what you see in the book. An excellent starting place is the Web Guide for Discrete Mathematics that can be found on the Web site for this book. The URL is http://www.mhhe.com/rosen.

Kenneth H. Rosen

CONTENTS

Preface ix

The Companion Web Site xix

To the Student xxi

1

The Foundations: Logic, Sets, and Functions 1

1.1 Logic 1
1.2 Propositional Equivalences 14
1.3 Predicates and Quantifiers 21
1.4 Sets 38
1.5 Set Operations 46
1.6 Functions 56
1.7 Sequences and Summations 69
1.8 The Growth of Functions 80

Key Terms and Results 92
Review Questions 94
Supplementary Exercises 95
Computer Projects 97
Computations and Explorations 97
Writing Projects 97

2

The Fundamentals: Algorithms, the Integers, and Matrices 99

2.1 Algorithms 99
2.2 Complexity of Algorithms 105
2.3 The Integers and Division 112
2.4 Integers and Algorithms 127
2.5 Applications of Number Theory 137
2.6 Matrices 150

Key Terms and Results 161
Review Questions 162
Supplementary Exercises 163

Computer Projects 164
Computations and Explorations 165
Writing Projects 165

3
Mathematical Reasoning 167

3.1 Methods of Proof 167
3.2 Mathematical Induction 186
3.3 Recursive Definitions 202
3.4 Recursive Algorithms 214
3.5 Program Correctness 219

Key Terms and Results 225
Review Questions 226
Supplementary Exercises 227
Computer Projects 229
Computations and Explorations 230
Writing Projects 230

4
Counting 232

4.1 The Basics of Counting 232
4.2 The Pigeonhole Principle 244
4.3 Permutations and Combinations 250
4.4 Discrete Probability 260
4.5 Probability Theory 267
4.6 Generalized Permutations and Combinations 286
4.7 Generating Permutations and Combinations 296

Key Terms and Concepts 301
Review Questions 302
Supplementary Exercises 303
Computer Projects 306
Computations and Explorations 306
Writing Projects 307

5
Advanced Counting Techniques 308

5.1 Recurrence Relations 308
5.2 Solving Recurrence Relations 319
5.3 Divide-and-Conquer Relations 332
5.4 Generating Functions 338
5.5 Inclusion–Exclusion 354
5.6 Applications of Inclusion–Exclusion 360

Key Terms and Results 368
Review Questions 369

Supplementary Exercises 369
Computer Projects 371
Computations and Explorations 372
Writing Projects 372

6

Relations 374

6.1 Relations and Their Properties 374
6.2 *n*-ary Relations and Their Applications 384
6.3 Representing Relations 390
6.4 Closures of Relations 396
6.5 Equivalence Relations 408
6.6 Partial Orderings 415

Key Terms and Results 430
Review Questions 431
Supplementary Exercises 432
Computer Projects 436
Computations and Explorations 436
Writing Projects 436

7

Graphs 438

7.1 Introduction to Graphs 438
7.2 Graph Terminology 445
7.3 Representing Graphs and Graph Isomorphism 456
7.4 Connectivity 467
7.5 Euler and Hamilton Paths 475
7.6 Shortest Path Problems 490
7.7 Planar Graphs 501
7.8 Graph Coloring 510

Key Terms and Results 519
Review Questions 521
Supplementary Exercises 522
Computer Projects 525
Computations and Explorations 526
Writing Projects 527

8

Trees 528

8.1 Introduction to Trees 528
8.2 Applications of Trees 541
8.3 Tree Traversal 547
8.4 Trees and Sorting 562
8.5 Spanning Trees 570
8.6 Minimum Spanning Trees 580

Key Terms and Results 587
Review Questions 588
Supplementary Exercises 588
Computer Projects 591
Computations and Explorations 591
Writing Projects 592

9
Boolean Algebra 593

9.1 Boolean Functions 593
9.2 Representing Boolean Functions 600
9.3 Logic Gates 604
9.4 Minimization of Circuits 611

Key Terms and Results 625
Review Questions 625
Supplementary Exercises 626
Computer Projects 627
Computations and Explorations 628
Writing Projects 628

10
Modeling Computation 629

10.1 Languages and Grammars 629
10.2 Finite-State Machines with Output 640
10.3 Finite-State Machines with No Output 647
10.4 Language Recognition 656
10.5 Turing Machines 666

Key Terms and Results 674
Review Questions 675
Supplementary Exercises 675
Computer Projects 677
Computations and Explorations 678
Writing Projects 678

Appendixes A-1

A.1 Exponential and Logarithmic Functions A-1
A.2 Pseudocode A-5

Suggested Readings B-1

Index of Biographies I-1

Index I-3

LIST OF SYMBOLS L-1

The Foundations: Logic, Sets, and Functions

1

This chapter reviews the foundations of discrete mathematics. Three important topics are covered: logic, sets, and functions. The rules of logic specify the precise meaning of mathematical statements. For instance, the rules give us the meaning of such statements as, "There exists an integer that is greater than 100 that is a power of 2," and, "For every integer n the sum of the positive integers not exceeding n is $n(n + 1)/2$." Logic is the basis of all mathematical reasoning, and it has practical applications to the design of computing machines, to artificial intelligence, to computer programming, to programming languages, and to other areas of computer science.

Much of discrete mathematics is devoted to the study of discrete structures, which are used to represent discrete objects. All discrete structures are built up from sets, which are collections of objects. Examples of discrete structures built up from sets include combinations, which are unordered collections of objects used extensively in counting; relations, which are sets of ordered pairs that represent relationships between objects; graphs, which are sets of vertices and edges that connect vertices; and finite state machines, which are used to model computing machines.

The concept of a function is extremely important in discrete mathematics. A function assigns to each element of a set precisely one element of a set. Such useful structures as sequences and strings are special types of functions. Functions are used to represent the number of steps a procedure uses to solve a problem. The analysis of algorithms uses terminology and concepts related to the growth of functions. Recursive functions, defined by specifying their values at positive integers in terms of their values at smaller positive integers, are used to solve many counting problems.

1.1

Logic

INTRODUCTION

The rules of logic give precise meaning to mathematical statements. These rules are used to distinguish between valid and invalid mathematical arguments. Since a major goal of this book is to teach the reader how to understand and how to construct correct mathematical arguments, we begin our study of discrete mathematics with an introduction to logic.

In addition to its importance in understanding mathematical reasoning, logic has numerous applications in computer science. These rules are used in the design of computer circuits, the construction of computer programs, the verification of the correctness of programs, and in many other ways. We will discuss each of these applications in the following chapters.

PROPOSITIONS

Our discussion begins with an introduction to the basic building blocks of logic—propositions. A **proposition** is a statement that is either true or false, but not both.

EXAMPLE 1 All the following statements are propositions.

1. Washington, D.C., is the capital of the United States of America.
2. Toronto is the capital of Canada.
3. $1 + 1 = 2$.
4. $2 + 2 = 3$.

Propositions 1 and 3 are true, whereas 2 and 4 are false. ■

Some sentences that are not propositions are given in the next example.

EXAMPLE 2 Consider the following sentences.

1. What time is it?
2. Read this carefully.
3. $x + 1 = 2$.
4. $x + y = z$.

Sentences 1 and 2 are not propositions because they are not statements. Sentences 3 and 4 are not propositions because they are neither true nor false, since the variables in these sentences have not been assigned values. Various ways to form propositions from sentences of this type will be discussed in Section 1.3. ■

Letters are used to denote propositions, just as letters are used to denote variables. The conventional letters used for this purpose are $p, q, r, s, \ldots$. The **truth value** of a proposition is true, denoted by T, if it is a true proposition and false, denoted by F, if it is a false proposition.

web

We now turn our attention to methods for producing new propositions from those that we already have. These methods were discussed by the English mathematician George Boole in 1854 in his book *The Laws of Thought.* Many mathematical statements are constructed by combining one or more propositions. New propositions, called **compound propositions,** are formed from existing propositions using logical operators.

DEFINITION 1. Let p be a proposition. The statement

"It is not the case that p."

is another proposition, called the *negation* of p. The negation of p is denoted by $\neg p$. The proposition $\neg p$ is read "not p."

EXAMPLE 3 Find the negation of the proposition

"Today is Friday"

and express this in simple English.

Solution: The negation is

"It is not the case that today is Friday."

This negation can be more simply expressed by

"Today is not Friday." ■

Remark: Strictly speaking, sentences involving variable times such as those in Example 3 are not propositions unless a fixed time is assumed. The same holds for variable places unless a fixed place is assumed and for pronouns unless a particular person is assumed.

A **truth table** displays the relationships between the truth values of propositions. Truth tables are especially valuable in the determination of the truth values of propositions constructed from simpler propositions. Table 1 displays all possible truth values of a proposition and the corresponding truth values of its negation.

The negation of a proposition can also be considered the result of the operation of the **negation operator** on a proposition. The negation operator constructs a new proposition from a single existing proposition. We will now introduce the logical operators that are used to form new propositions from two or more existing propositions. These logical operators are also called **connectives.**

TABLE 1 The Truth Table for the Negation of a Proposition.

p	$\neg p$
T	F
F	T

---

web **George Boole (1815–1864).** George Boole, the son of a cobbler, was born in Lincoln, England, in November 1815. Because of his family's difficult financial situation, Boole had to struggle to educate himself while supporting his family. Nevertheless, he became one of the most important mathematicians of the 1800s. Although he considered a career as a clergyman, he decided instead to go into teaching and soon afterward opened a school of his own. In his preparation for teaching mathematics, Boole—unsatisfied with textbooks of his day—decided to read the works of the great mathematicians. While reading papers of the great French mathematician Lagrange, Boole made discoveries in the calculus of variations, the branch of analysis dealing with finding curves and surfaces optimizing certain parameters.

In 1848 Boole published *The Mathematical Analysis of Logic*, the first of his contributions to symbolic logic. In 1849 he was appointed professor of mathematics at Queen's College in Cork, Ireland. In 1854 he published *The Laws of Thought*, his most famous work. In this book Boole introduced what is now called *Boolean algebra* in his honor. Boole wrote textbooks on differential equations and on difference equations that were used in Great Britain until the end of the nineteenth century. Boole married in 1855; his wife was the niece of the professor of Greek at Queen's College. In 1864 Boole died from pneumonia, which he contracted as a result of keeping a lecture engagement even though he was soaking wet from a rainstorm.

DEFINITION 2. **Let p and q be propositions. The proposition "p and q," denoted by $p \wedge q$, is the proposition that is true when both p and q are true and is false otherwise. The proposition $p \wedge q$ is called the *conjunction* of p and q.**

The truth table for $p \wedge q$ is shown in Table 2. Note that there are four rows in this truth table, one row for each possible combination of truth values for the propositions p and q.

EXAMPLE 4 Find the conjunction of the propositions p and q where p is the proposition "Today is Friday" and q is the proposition "It is raining today."

Solution: The conjunction of these propositions, $p \wedge q$, is the proposition "Today is Friday and it is raining today." This proposition is true on rainy Fridays and is false on any day that is not a Friday and on Fridays when it does not rain. ■

DEFINITION 3. **Let p and q be propositions. The proposition "p or q," denoted by $p \vee q$, is the proposition that is false when p and q are both false and true otherwise. The proposition $p \vee q$ is called the *disjunction* of p and q.**

The truth table for $p \vee q$ is shown in Table 3.

The use of the connective *or* in a disjunction corresponds to one of the two ways the word *or* is used in English, namely, in an inclusive way. A disjunction is true when either of the two propositions in it is true or when both are true. For instance, the inclusive or is being used in the statement

> "Students who have taken calculus or computer science can take this class."

Here, we mean that students who have taken both calculus and computer science can take the class, as well as the students who have taken just one of the two subjects. On the other hand, we are using the exclusive or when we say

> "Students who have taken calculus or computer science, but not both, can enroll in this class."

Here, we mean that students who have taken both calculus and a computer science course cannot take the class. Only those who have taken exactly one of the two courses can take the class.

TABLE 2 The Truth Table for the Conjunction of Two Propositions.

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

TABLE 3 The Truth Table for the Disjunction of Two Propositions.

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

Similarly, when a menu at a restaurant states, "Soup or salad comes with an entrée," the restaurant almost always means that customers can have either soup or salad, but not both. Hence, this is an exclusive, rather than an inclusive, or.

EXAMPLE 5 What is the disjunction of the propositions p and q where p and q are the same propositions as in Example 4?

Solution: The disjunction of p and q, $p \vee q$, is the proposition

"Today is Friday or it is raining today."

This proposition is true on any day that is either a Friday or a rainy day (including rainy Fridays). It is only false on days that are not Fridays when it also does not rain. ■

As was previously remarked, the use of the connective or in a disjunction corresponds to one of the two ways the word *or* is used in English, namely, in an inclusive way. Thus, a disjunction is true when either of the two propositions in it is true or when both are true. Sometimes, we use *or* in an exclusive sense. When the exclusive or is used to connect the propositions p and q, the proposition "p or q (but not both)" is obtained. This proposition is true when p is true and q is false, or vice versa, and it is false when both p and q are false and when both are true.

DEFINITION 4. Let p and q be propositions. The *exclusive or* of p and q, denoted by $p \oplus q$, is the proposition that is true when exactly one of p and q is true and is false otherwise.

The truth table for the exclusive or of two propositions is displayed in Table 4.

We will discuss several other important ways that propositions may be combined.

DEFINITION 5. Let p and q be propositions. The *implication* $p \rightarrow q$ is the proposition that is false when p is true and q is false and true otherwise. In this implication p is called the *hypothesis* (or *antecedent* or *premise*) and q is called the *conclusion* (or *consequence*).

The truth table for the implication $p \rightarrow q$ is shown in Table 5.

TABLE 4 The Truth Table for the Exclusive Or of Two Propositions.

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

TABLE 5 The Truth Table for the Implication $p \rightarrow q$.

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Because implications arise in many places in mathematical reasoning, a wide variety of terminology is used to express $p \to q$. Some of the more common ways of expressing this implication are:

- "if p, then q"
- "p implies q"
- "if p, q"
- "p only if q"
- "p is sufficient for q"
- "q if p"
- "q whenever p"
- "q is necessary for p."

Note that $p \to q$ is false only in the case that p is true but q is false, so that it is true when both p and q are true, and when p is false (no matter what truth value q has).

A useful way to remember that an implication is true when its hypothesis is false is to think of a contract or an obligation. If the condition specified by such a statement is false, no obligation is in force. For example, the statement "If you make more than \$25,000, then you must file a tax return" says nothing about someone making less than \$25,000. You violate the obligation only if you make more than \$25,000 and do not file a return. Similarly, the statement "If a player hits more than 60 home runs, then a bonus of \$10 million is awarded" in the contract of a baseball player is violated only when the player hits more than 60 home runs, but the bonus is not awarded. This says nothing if the player hits fewer than 60 home runs.

The way we have defined implications is more general than the meaning attached to implications in the English language. For instance, the implication

"If it is sunny today, then we will go to the beach."

is an implication used in normal language, since there is a relationship between the hypothesis and the conclusion. Further, this implication is considered valid unless it is indeed sunny today, but we do not go to the beach. On the other hand, the implication

"If today is Friday, then $2 + 3 = 5$."

is true from the definition of implication, since its conclusion is true. (The truth value of the hypothesis does not matter then.) The implication

"If today is Friday, then $2 + 3 = 6$."

is true every day except Friday, even though $2 + 3 = 6$ is false.

We would not use these last two implications in natural language, since there is no relationship between the hypothesis and the conclusion in either implication. In mathematical reasoning we consider implications of a more general sort than we use in English. The mathematical concept of an implication is independent of a cause-and-effect relationship between hypothesis and conclusion. Our definition of an implication specifies its truth values; it is not based on English usage.

The if-then construction used in many programming languages is different from that used in logic. Most programming languages contain statements such as **if** p **then** S, where p is a proposition and S is a program segment (one or more statements to be executed). When execution of a program encounters such a statement, S is executed if p is true, but S is not executed if p is false, as illustrated in the following example.

EXAMPLE 6 What is the value of the variable x after the statement

if $2 + 2 = 4$ **then** $x := x + 1$

if $x = 0$ before this statement is encountered? (The symbol $:=$ stands for assignment. The statement $x := x + 1$ means the assignment of the value of $x + 1$ to x.)

Solution: Since $2 + 2 = 4$ is true, the assignment statement $x := x + 1$ is executed. Hence, x has the value $0 + 1 = 1$ after this statement is encountered. ■

We can build up compound propositions using the negation operator and the different connectives defined so far. Parentheses are used to specify the order in which the various logical operators in a compound proposition are applied. In particular, the logical operators in the innermost parentheses are applied first. For instance, $(p \vee q) \wedge (\neg r)$ is the conjunction of $p \vee q$ and $\neg r$. To cut down on the number of parentheses needed, we specify that the negation operator is applied before all other logical operators. This means that $\neg p \wedge q$ is the conjunction of $\neg p$ and q, namely $(\neg p) \wedge q$, not the negation of the conjunction of p and q, namely $\neg(p \wedge q)$.

There are some related implications that can be formed from $p \rightarrow q$. The proposition $q \rightarrow p$ is called the **converse** of $p \rightarrow q$. The **contrapositive** of $p \rightarrow q$ is the proposition $\neg q \rightarrow \neg p$.

EXAMPLE 7 Find the converse and the contrapositive of the implication

"If today is Thursday, then I have a test today."

Solution: The converse is

"If I have a test today, then today is Thursday."

And the contrapositive of this implication is

"If I do not have a test today, then today is not Thursday." ■

We now introduce another way to combine propositions.

DEFINITION 6. Let p and q be propositions. The *biconditional* $p \leftrightarrow q$ is the proposition that is true when p and q have the same truth values and is false otherwise.

The truth table for $p \leftrightarrow q$ is shown in Table 6. Note that the biconditional $p \leftrightarrow q$ is true precisely when both the implications $p \rightarrow q$ and $q \rightarrow p$ are true. Because of this, the terminology

"p if and only if q"

is used for this biconditional. Other common ways of expressing the proposition $p \leftrightarrow q$ are: "p is necessary and sufficient for q" and "if p then q, and conversely."

TABLE 6 The Truth Table for the Biconditional $p \leftrightarrow q$.

p	q	$p \leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

TRANSLATING ENGLISH SENTENCES

There are many reasons to translate English sentences into expressions involving propositional variables and logical connectives. In particular, English (and every other human language) is often ambiguous. Translating sentences into logical expressions removes the ambiguity. Note that this may involve making a set of reasonable assumptions based on the intended meaning of the sentence. Moreover, once we have translated sentences from English into logical expressions we can analyze these logical expressions to determine their truth values, we can manipulate them, and we can use rules of inference (which are discussed in Chapter 3) to reason about them.

To illustrate the process of translating an English sentence into a logical expression, consider the following examples.

EXAMPLE 8 How can the following English sentence be translated into a logical expression?

> "You can access the Internet from campus only if you are a computer science major or you are not a freshman."

Solution: There are many ways to translate this sentence into a logical expression. Although it is possible to represent the sentence by a single propositional variable, such as p, this would not be useful when analyzing its meaning or reasoning with it. Instead, we will use propositional variables to represent each sentence part and determine the appropriate logical connectives between them. In particular, we let a, c, and f represent "You can access the Internet from campus," "You are a computer science major," and "You are a freshman," respectively. Noting that "only if" is one way an implication can be expressed, this sentence can be represented as

$$a \to (c \lor \neg f).$$

■

EXAMPLE 9 How can the following English sentence be translated into a logical expression?

> "You cannot ride the roller coaster if you are under 4 feet tall unless you are older than 16 years old."

Solution: There are many ways to translate this sentence into a logical expression. The simplest but least useful way is simply to represent the sentence by a single propositional variable, say, p. Although this is not wrong, doing this would not assist us when we try to analyze the sentence or reason using it. More appropriately, what we can do is to use propositional variables to represent each of the sentence parts and to decide on the appropriate logical connectives between them. In particular, we let q, r, and s represent "You can ride the roller coaster," "You are under 4 feet tall," and "You are older than 16 years old," respectively. Then the sentence can be translated to

$$(r \land \neg s) \to \neg q.$$

Of course, there are other ways to represent the original sentence as a logical expression, but the one we have used should meet our needs. ■

BOOLEAN SEARCHES

web

Logical connectives are used extensively in searches of large collections of information, such as indexes of Web pages. Because these searches employ techniques from propositional logic, they are called **Boolean searches.**

In Boolean searches, the connective *AND* is used to match records that contain both of two search terms, the connective *OR* is used to match one or both of two search terms, and the connective *NOT* (sometimes written as *AND NOT*) is used to exclude a particular search term. Careful planning of how logical connectives are used is often required when Boolean searches are used to locate information of potential interest. The following example illustrates how Boolean searches are carried out.

EXAMPLE 10 Web Page Searching. Most Web search engines support Boolean searching techniques, which usually can help find Web pages about particular subjects. For instance, using Boolean searching to find Web pages about universities in New Mexico, we can look for pages matching NEW *AND* MEXICO *AND* UNIVERSITIES. The results of this search will include those pages that contain the three words NEW, MEXICO, and UNIVERSITIES. This will include all of the pages of interest, together with others such as a page about new universities in Mexico. Next, to find pages that deal with universities in New Mexico or Arizona, we can search for pages matching (NEW *AND* MEXICO *OR* ARIZONA) *AND* UNIVERSITIES. (*Note:* Here the *AND* operator takes precedence over the *OR* operator.) The results of this search will include all pages that contain the word UNIVERSITIES and either both the words NEW and MEXICO or the word ARIZONA. Again, pages besides those of interest will be listed. Finally, to find Web pages that deal with universities in Mexico (and not New Mexico), we might first look for pages matching MEXICO *AND* UNIVERSITIES, but since the results of this search will include pages about universities in New Mexico, as well as universities in Mexico, it might be better to search for pages matching (MEXICO *AND* UNIVERSITIES) *NOT* NEW. The results of this search include pages that contain both the words MEXICO and UNIVERSITIES but do not contain the word NEW. ■

LOGIC AND BIT OPERATIONS

web

Computers represent information using bits. A **bit** has two possible values, namely, 0 (zero) and 1 (one). This meaning of the word bit comes from *b*inary dig*it*, since zeros and ones are the digits used in binary representations of numbers. The well-known statistician John Tukey introduced this terminology in 1946. A bit can be used to represent a truth value, since there are two truth values, namely, *true* and *false*. As is customarily done, we will use a 1 bit to represent true and a 0 bit to represent false. That is, 1 represents T (true), 0 represents F (false). A variable is called a **Boolean variable** if its value is either true or false. Consequently, a Boolean variable can be represented using a bit.

Computer **bit operations** correspond to the logical connectives. By replacing true by a one and false by a zero in the truth tables for the operators $\wedge$, $\vee$, and $\oplus$, the tables shown in Table 7 for the corresponding bit operations are obtained. We will also use the notation *OR*, *AND*, and *XOR* for the operators $\vee$, $\wedge$, and $\oplus$, as is done in various programming languages.

Information is often represented using bit strings, which are sequences of zeros and ones. When this is done, operations on the bit strings can be used to manipulate this information.

DEFINITION 7. **A *bit string* is a sequence of zero or more bits. The *length* of this string is the number of bits in the string.**

TABLE 7 **Tables for the Bit Operators *OR*, *AND*, and *XOR*.**

x	y	$x \vee y$	$x \wedge y$	$x \oplus y$
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

EXAMPLE 11 101010011 is a bit string of length nine. ■

We can extend bit operations to bit strings. We define the **bitwise *OR*, bitwise *AND*,** and **bitwise *XOR*** of two strings of the same length to be the strings that have as their bits the *OR*, *AND*, and *XOR* of the corresponding bits in the two strings, respectively. We use the symbols $\vee$, $\wedge$, and $\oplus$ to represent the bitwise *OR*, bitwise *AND*, and bitwise *XOR* operations, respectively. We illustrate bitwise operations on bit strings with the following example.

EXAMPLE 12 Find the bitwise *OR*, bitwise *AND*, and bitwise *XOR* of the bit strings 01 1011 0110 and 11 0001 1101. (Here, and throughout this book, bit strings will be split into blocks of four bits to make them easier to read.)

Solution: The bitwise *OR*, bitwise *AND*, and bitwise *XOR* of these strings are obtained by taking the *OR*, *AND*, and *XOR* of the corresponding bits, respectively. This gives us

---

Historical Note: There were several other suggested words for a binary digit, including *binit* and *bigit*, that never were widely accepted. The adoption of the word *bit* may be due to its meaning as a common English word. For an account of Tukey's coining of the word *bit*, see the April 1984 issue of *Annals of the History of Computing*.

web

John Wilder Tukey (born 1915). Tukey, born in New Bedford, Massachusetts, was an only child. His parents, both teachers, decided home schooling would best develop his potential. His formal education began at Brown University, where he studied mathematics and chemistry. He received a master's degree in chemistry from Brown and continued his studies at Princeton University, changing his field of study from chemistry to mathematics. He received his Ph.D. from Princeton in 1939 for work in topology, when he was appointed an instructor in mathematics at Princeton. With the start of World War II, he joined the Fire Control Research Office, where he began working in statistics. Tukey found statistical research to his liking and impressed several leading statisticians with his skills. In 1945, at the conclusion of the war, Tukey returned to the mathematics department at Princeton as a professor of statistics, and he also took a position at AT&T Bell Laboratories. Tukey founded the Statistics Department at Princeton in 1966 and was its first chairman. Tukey made significant contributions to many areas of statistics, including the analysis of variance, the estimation of spectra of time series, inferences about the values of a set of parameters from a single experiment, and the philosophy of statistics. However, he is best known for his invention, with J. W. Cooley, of the fast Fourier transform.

Tukey contributed his insight and expertise by serving on the President's Science Advisory Committee. He chaired several important committees dealing with the environment, education, and chemicals and health. He also served on committees working on nuclear disarmament. Tukey has received many awards, including the National Medal of Science.

01 1011 0110	
11 0001 1101	
11 1011 1111	bitwise *OR*
01 0001 0100	bitwise *AND*
10 1010 1011	bitwise *XOR*

■

Exercises

1. Which of the following sentences are propositions? What are the truth values of those that are propositions?
- **a)** Boston is the capital of Massachusetts.
- **b)** Miami is the capital of Florida.
- **c)** $2 + 3 = 5$.
- **d)** $5 + 7 = 10$.
- **e)** $x + 2 = 11$.
- **f)** Answer this question.
- **g)** $x + y = y + x$ for every pair of real numbers x and y.

2. Which of the following are propositions? What are the truth values of those that are propositions?
- **a)** Do not pass go.
- **b)** What time is it?
- **c)** There are no black flies in Maine.
- **d)** $4 + x = 5$.
- **e)** $x + 1 = 5$ if $x = 1$.
- **f)** $x + y = y + z$ if $x = z$.

3. What is the negation of each of the following propositions?
- **a)** Today is Thursday.
- **b)** There is no pollution in New Jersey.
- **c)** $2 + 1 = 3$.
- **d)** The summer in Maine is hot and sunny.

4. Let p and q be the propositions

p : I bought a lottery ticket this week.
q : I won the million dollar jackpot on Friday.

Express each of the following propositions as an English sentence.
- **a)** $\neg p$ **b)** $p \vee q$
- **c)** $p \rightarrow q$ **d)** $p \wedge q$
- **e)** $p \leftrightarrow q$ **f)** $\neg p \rightarrow \neg q$
- **g)** $\neg p \wedge \neg q$ **h)** $\neg p \vee (p \wedge q)$

5. Let p and q be the propositions

p : It is below freezing.
q : It is snowing.

Write the following propositions using p and q and logical connectives.
- **a)** It is below freezing and snowing.
- **b)** It is below freezing but not snowing.
- **c)** It is not below freezing and it is not snowing.
- **d)** It is either snowing or below freezing (or both).
- **e)** If it is below freezing, it is also snowing.
- **f)** It is either below freezing or it is snowing, but it is not snowing if it is below freezing.
- **g)** That it is below freezing is necessary and sufficient for it to be snowing.

6. Let p, q, and r be the propositions

p : You have the flu.
q : You miss the final examination.
r : You pass the course.

Express each of the following propositions as an English sentence.
- **a)** $p \rightarrow q$ **b)** $\neg q \leftrightarrow r$
- **c)** $q \rightarrow \neg r$ **d)** $p \vee q \vee r$
- **e)** $(p \rightarrow \neg r) \vee (q \rightarrow \neg r)$
- **f)** $(p \wedge q) \vee (\neg q \wedge r)$

7. Let p and q be the propositions

p : You drive over 65 miles per hour.
q : You get a speeding ticket.

Write the following propositions using p and q and logical connectives.
- **a)** You do not drive over 65 miles per hour.
- **b)** You drive over 65 miles per hour, but you do not get a speeding ticket.
- **c)** You will get a speeding ticket if you drive over 65 miles per hour.
- **d)** If you do not drive over 65 miles per hour, then you will not get a speeding ticket.
- **e)** Driving over 65 miles per hour is sufficient for getting a speeding ticket.
- **f)** You get a speeding ticket, but you do not drive over 65 miles per hour.
- **g)** Whenever you get a speeding ticket, you are driving over 65 miles per hour.

8. Let p, q, and r be the propositions

p : You get an A on the final exam.
q : You do every exercise in this book.
r : You get an A in this class.

Write the following propositions using p, q, and r and logical connectives.

- **a)** You get an A in this class, but you do not do every exercise in this book.
- **b)** You get an A on the final, you do every exercise in this book, and you get an A in this class.
- **c)** To get an A in this class, it is necessary for you to get an A on the final.
- **d)** You get an A on the final, but you don't do every exercise in this book; nevertheless, you get an A in this class.
- **e)** Getting an A on the final and doing every exercise in this book is sufficient for getting an A in this class.
- **f)** You will get an A in this class if and only if you either do every exercise in this book or you get an A on the final.

9. Determine whether each of the following implications is true or false.

- **a)** If $1 + 1 = 2$, then $2 + 2 = 5$.
- **b)** If $1 + 1 = 3$, then $2 + 2 = 4$.
- **c)** If $1 + 1 = 3$, then $2 + 2 = 5$.
- **d)** If pigs can fly, then $1 + 1 = 3$.
- **e)** If $1 + 1 = 3$, then God exists.
- **f)** If $1 + 1 = 3$, then pigs can fly.
- **g)** If $1 + 1 = 2$, then pigs can fly.
- **h)** If $2 + 2 = 4$, then $1 + 2 = 3$.

10. For each of the following sentences, determine whether an inclusive or or an exclusive or is intended. Explain your answer.

- **a)** Experience with C++ or Java is required.
- **b)** Lunch includes soup or salad.
- **c)** To enter the country you need a passport or a voter registration card.
- **d)** Publish or perish.

11. For each of the following sentences, state what the sentence means if the or is an inclusive or (that is, a disjunction) versus an exclusive or. Which of these meanings of or do you think is intended?

- **a)** To take discrete mathematics, you must have taken calculus or a course in computer science.
- **b)** When you buy a new car from Acme Motor Company, you get \$2000 back in cash or a 2% car loan.
- **c)** Dinner for two includes two items from column A or three items from column B.
- **d)** School is closed if more than 2 feet of snow falls or if the wind chill is below -100.

12. An ancient Sicilian legend says that the barber in a remote town who can be reached only by traveling a dangerous mountain road shaves those people, and only those people, who do not shave themselves. Can there be such a barber?

13. Each inhabitant of a remote village always tells the truth or always lies. A villager will only give a "Yes" or a "No" response to a question a tourist asks. Suppose you are a tourist visiting this area and come to a fork in the road. One branch leads to the ruins you want to visit; the other branch leads deep into the jungle. A villager is standing at the fork in the road. What one question can you ask the villager to determine which branch to take?

14. An explorer is captured by a group of cannibals. There are two types of cannibals—those who always tell the truth and those who always lie. The cannibals will barbecue the explorer unless he can determine whether a particular cannibal always lies or always tells the truth. He is allowed to ask the cannibal exactly one question.

- **a)** Explain why the question "Are you a liar?" does not work.
- **b)** Find a question that the explorer can use to determine whether the cannibal always lies or always tells the truth.

15. Write each of the following statements in the form "if p, then q" in English. (*Hint:* Refer to the list of common ways to express implications listed in this section.)

- **a)** It snows whenever the wind blows from the northeast.
- **b)** The apple trees will bloom if it stays warm for a week.
- **c)** That the Pistons win the championship implies that they beat the Lakers.
- **d)** It is necessary to walk 8 miles to get to the top of Long's Peak.
- **e)** To get tenure as a professor, it is sufficient to be world-famous.
- **f)** If you drive more than 400 miles, you will need to buy gasoline.
- **g)** Your guarantee is good only if you bought your CD player less than 90 days ago.

16. Write each of the following statements in the form "if p then q" in English. (*Hint:* Refer to the list of common ways to express implications listed in this section.)

- **a)** I will remember to send you the address only if you send me an e-mail message.
- **b)** To be a citizen of this country, it is sufficient that you were born in the United States.
- **c)** If you keep your textbook, it will be a useful reference in your future courses.
- **d)** The Red Wings will win the Stanley Cup if their goalie plays well.
- **e)** That you get the job implies that you had the best credentials.
- **f)** The beach erodes whenever there is a storm.
- **g)** It is necessary to have a valid password to log on to the server.

17. Write each of the following propositions in the form "p if and only if q" in English.

- **a)** If it is hot outside you buy an ice cream cone, and if you buy an ice cream cone it is hot outside.
- **b)** For you to win the contest it is necessary and sufficient that you have the only winning ticket.
- **c)** You get promoted only if you have connections, and you have connections only if you get promoted.

d) If you watch television your mind will decay, and conversely.

e) The trains run late on exactly those days when I take it.

18. Write each of the following propositions in the form "p if and only if q" in English.

a) For you to get an A in this course, it is necessary and sufficient that you learn how to solve discrete mathematics problems.

b) If you read the newspaper every day, you will be informed, and conversely.

c) It rains if it is a weekend day, and it is a weekend day if it rains.

d) You can see the wizard only if the wizard is not in, and the wizard is not in only if you can see him.

19. State the converse and contrapositive of each of the following implications.

a) If it snows today, I will ski tomorrow.

b) I come to class whenever there is going to be a quiz.

c) A positive integer is a prime only if it has no divisors other than 1 and itself.

20. State the converse and contrapositive of each of the following implications.

a) If it snows tonight, then I will stay at home.

b) I go to the beach whenever it is a sunny summer day.

c) When I stay up late, it is necessary that I sleep until noon.

21. Construct a truth table for each of the following compound propositions.

a) $p \wedge \neg p$ **b)** $p \vee \neg p$

c) $(p \vee \neg q) \rightarrow q$ **d)** $(p \vee q) \rightarrow (p \wedge q)$

e) $(p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)$

f) $(p \rightarrow q) \rightarrow (q \rightarrow p)$

22. Construct a truth table for each of the following compound propositions.

a) $p \oplus p$ **b)** $p \oplus \neg p$

c) $p \oplus \neg q$ **d)** $\neg p \oplus \neg q$

e) $(p \oplus q) \vee (p \oplus \neg q)$ **f)** $(p \oplus q) \wedge (p \oplus \neg q)$

23. Construct a truth table for each of the following compound propositions.

a) $p \rightarrow \neg q$

b) $\neg p \leftrightarrow q$

c) $(p \rightarrow q) \vee (\neg p \rightarrow q)$

d) $(p \rightarrow q) \wedge (\neg p \rightarrow q)$

e) $(p \leftrightarrow q) \vee (\neg p \leftrightarrow q)$

f) $(\neg p \leftrightarrow \neg q) \leftrightarrow (p \leftrightarrow q)$

24. Construct a truth table for each of the following compound propositions.

a) $(p \vee q) \vee r$ **b)** $(p \vee q) \wedge r$

c) $(p \wedge q) \vee r$ **d)** $(p \wedge q) \wedge r$

e) $(p \vee q) \wedge \neg r$ **f)** $(p \wedge q) \vee \neg r$

25. Construct a truth table for each of the following compound propositions.

a) $p \rightarrow (\neg q \vee r)$

b) $\neg p \rightarrow (q \rightarrow r)$

c) $(p \rightarrow q) \vee (\neg p \rightarrow r)$

d) $(p \rightarrow q) \wedge (\neg p \rightarrow r)$

e) $(p \leftrightarrow q) \vee (\neg q \leftrightarrow r)$

f) $(\neg p \leftrightarrow \neg q) \leftrightarrow (q \leftrightarrow r)$

26. Construct a truth table for $((p \rightarrow q) \rightarrow r) \rightarrow s$.

27. Construct a truth table for $(p \leftrightarrow q) \leftrightarrow (r \leftrightarrow s)$.

28. What is the value of x after each of the following statements is encountered in a computer program, if $x = 1$ before the statement is reached?

a) **if** $1 + 2 = 3$ **then** $x := x + 1$

b) **if** $(1 + 1 = 3)$ *OR* $(2 + 2 = 3)$ **then** $x := x + 1$

c) **if** $(2 + 3 = 5)$ *AND* $(3 + 4 = 7)$ **then** $x := x + 1$

d) **if** $(1 + 1 = 2)$ *XOR* $(1 + 2 = 3)$ **then** $x := x + 1$

e) **if** $x < 2$ **then** $x := x + 1$

29. Find the bitwise *OR*, bitwise *AND*, and bitwise *XOR* of each of the following pairs of bit strings.

a) 101 1110, 010 0001

b) 1111 0000, 1010 1010

c) 00 0111 0001, 10 0100 1000

d) 11 1111 1111, 00 0000 0000

30. Evaluate each of the following expressions.

a) $1\,1000 \wedge (0\,1011 \vee 1\,1011)$

b) $(0\,1111 \wedge 1\,0101) \vee 0\,1000$

c) $(0\,1010 \oplus 1\,1011) \oplus 0\,1000$

d) $(1\,1011 \vee 0\,1010) \wedge (1\,0001 \vee 1\,1011)$

web **Fuzzy logic** is used in artificial intelligence. In fuzzy logic, a proposition has a truth value that is a number between 0 and 1, inclusive. A proposition with a truth value of 0 is false and one with a truth value of 1 is true. Truth values that are between 0 and 1 indicate varying degrees of truth. For instance, the truth value 0.8 can be assigned to the statement "Fred is happy," since Fred is happy most of the time, and the truth value 0.4 can be assigned to the statement "John is happy," since John is happy slightly less than half the time.

31. The truth value of the negation of a proposition in fuzzy logic is 1 minus the truth value of the proposition. What are the truth values of the statements "Fred is not happy" and "John is not happy"?

32. The truth value of the conjunction of two propositions in fuzzy logic is the minimum of the truth values of the two propositions. What are the truth values of the statements "Fred and John are happy" and "Neither Fred nor John is happy"?

33. The truth value of the disjunction of two propositions in fuzzy logic is the maximum of the truth values of the two propositions. What are the truth values of the statements "Fred is happy, or John is happy" and "Fred is not happy, or John is not happy"?

***34.** Is the assertion "This statement is false" a proposition?

A set of propositional expressions is **consistent** if there is an assignment of truth values to the variables in the expressions that makes each expression true. When giving system specifications it is important that these specifications be consistent.

35. Are the following specifications consistent? "The system is in multiuser state if and only if it is operating normally. If the system is operating normally, the kernel is functioning. The kernel is not functioning or the system is in interrupt mode. If the system is not in multiuser state, then it is in interrupt mode. The system is not in interrupt mode."

36. Are the following specifications consistent? "If the file system is not locked, then new messages will be queued. If the file system is not locked, then the system is functioning normally, and conversely. If new messages are not queued, then they will be sent to the message buffer. If the file system is not locked, then new messages will be sent to the message buffer. New messages will not be sent to the message buffer."

37. What Boolean search would you use to look for Web pages about beaches in New Jersey? What if you wanted to find Web pages about beaches on the isle of Jersey (in the English Channel)?

38. What Boolean search would you use to look for Web pages about hiking in West Virginia? What if you wanted to find Web pages about hiking in Virginia, but not in West Virginia?

Exercises 39–42 are puzzles that can be solved by translating statements into logical expressions and reasoning from these expressions using truth tables.

39. Steve would like to determine the relative salaries of three coworkers using two facts. First, he knows that if Fred is not the highest paid of the three, then Janice is. Second, he knows that if Janice is not the lowest paid, then Maggie is paid the most. Is it possible to determine the relative salaries of Fred, Maggie, and Janice from what Steve knows? If so, who is paid the most and who the least? Explain your reasoning.

40. Five friends have access to a chat room. Is it possible to determine who is chatting if the following information is known? Either Kevin or Heather, or both, are chatting. Either Randy or Vijay, but not both, are chatting. If Abby is chatting, so is Randy. Vijay and Kevin are either both chatting or neither is. If Heather is chatting, then so are Abby and Kevin. Explain your reasoning.

41. A detective has interviewed four witnesses to a crime. From the stories of the witnesses the detective has concluded that if the butler is telling the truth then so is the cook; the cook and the gardener cannot both be telling the truth; the gardener and the handyman are not both lying; and if the handyman is telling the truth then the cook is lying. For each of the four witnesses, can the detective determine whether that person is telling the truth or lying? Explain your reasoning.

42. Four friends have been identified as suspects for an unauthorized access into a computer system. They have made statements to the investigating authorities. Alice said "Carlos did it." John said "I did not do it." Carlos said "Diana did it." Diana said "Carlos lied when he said that I did it."

a) If the authorities also know that exactly one of the four suspects is telling the truth, who did it? Explain your reasoning.

b) If the authorities also know that exactly one is lying, who did it? Explain your reasoning.

1.2

Propositional Equivalences

INTRODUCTION

An important type of step used in a mathematical argument is the replacement of a statement with another statement with the same truth value. Because of this, methods that produce propositions with the same truth value as a given compound proposition are used extensively in the construction of mathematical arguments.

We begin our discussion with a classification of compound propositions according to their possible truth values.

DEFINITION 1. A compound proposition that is always true, no matter what the truth values of the propositions that occur in it, is called a *tautology.* A compound proposition that is always false is called a *contradiction.* Finally, a proposition that is neither a tautology nor a contradiction is called a *contingency.*

Tautologies and contradictions are often important in mathematical reasoning. The following example illustrates these types of propositions.

TABLE 1 Examples of a Tautology and a Contradiction.

p	$\neg p$	$p \vee \neg p$	$p \wedge \neg p$
T	F	T	F
F	T	T	F

EXAMPLE 1 We can construct examples of tautologies and contradictions using just one proposition. Consider the truth tables of $p \vee \neg p$ and $p \wedge \neg p$, shown in Table 1. Since $p \vee \neg p$ is always true, it is a tautology. Since $p \wedge \neg p$ is always false, it is a contradiction. ■

LOGICAL EQUIVALENCES

Compound propositions that have the same truth values in all possible cases are called **logically equivalent.** We can also define this notion as follows.

DEFINITION 2. **The propositions p and q are called *logically equivalent* if $p \leftrightarrow q$ is a tautology. The notation $p \Longleftrightarrow q$ denotes that p and q are logically equivalent.**

One way to determine whether two propositions are equivalent is to use a truth table. In particular, the propositions p and q are equivalent if and only if the columns giving their truth values agree. The following example illustrates this method.

EXAMPLE 2 Show that $\neg(p \vee q)$ and $\neg p \wedge \neg q$ are logically equivalent. This equivalence is one of *De Morgan's laws* for propositions, named after the English mathematician Augustus De Morgan, of the mid-nineteenth century.

Solution: The truth tables for these propositions are displayed in Table 2. Since the truth values of the propositions $\neg(p \vee q)$ and $\neg p \wedge \neg q$ agree for all possible combinations of the truth values of p and q, it follows that these propositions are logically equivalent. ■

TABLE 2 Truth Tables for $\neg(p \vee q)$ and $\neg p \wedge \neg q$.

p	q	$p \vee q$	$\neg(p \vee q)$	$\neg p$	$\neg q$	$\neg p \wedge \neg q$
T	T	T	F	F	F	F
T	F	T	F	F	T	F
F	T	T	F	T	F	F
F	F	F	T	T	T	T

TABLE 3 **Truth Tables for $\neg p \vee q$ and $p \rightarrow q$.**

p	q	$\neg p$	$\neg p \vee q$	$p \rightarrow q$
T	T	F	T	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

EXAMPLE 3 Show that the propositions $p \rightarrow q$ and $\neg p \vee q$ are logically equivalent.

Solution: We construct the truth table for these propositions in Table 3. Since the truth values of $\neg p \vee q$ and $p \rightarrow q$ agree, these propositions are logically equivalent. ■

EXAMPLE 4 Show that the propositions $p \vee (q \wedge r)$ and $(p \vee q) \wedge (p \vee r)$ are logically equivalent. This is the *distributive law* of disjunction over conjunction.

Solution: We construct the truth table for these propositions in Table 4. Since the truth values of $p \vee (q \wedge r)$ and $(p \vee q) \wedge (p \vee r)$ agree, these propositions are logically equivalent. ■

Remark: A truth table of a compound proposition involving three different propositions requires eight rows, one for each possible combination of truth values of the three propositions. In general, 2^n rows are required if a compound proposition involves n propositions.

---

web

Augustus De Morgan (1806–1871). Augustus De Morgan was born in India, where his father was a colonel in the Indian army. De Morgan's family moved to England when he was 7 months old. He attended private schools, where he developed a strong interest in mathematics in his early teens. De Morgan studied at Trinity College, Cambridge, graduating in 1827. Although he considered entering medicine or law, he decided on a career in mathematics. He won a position at University College, London, in 1828, but resigned when the college dismissed a fellow professor without giving reasons. However, he resumed this position in 1836 when his successor died, staying there until 1866.

De Morgan was a noted teacher who stressed principles over techniques. His students included many famous mathematicians, including Ada Augusta, Countess of Lovelace, who was Charles Babbage's collaborator in his work on computing machines (see page 19 for biographical notes on Ada Augusta). (De Morgan cautioned the countess against studying too much mathematics, since it might interfere with her childbearing abilities!)

De Morgan was an extremely prolific writer. He wrote more than 1000 articles for more than 15 periodicals. De Morgan also wrote textbooks on many subjects, including logic, probability, calculus, and algebra. In 1838 he presented what was perhaps the first clear explanation of an important proof technique known as *mathematical induction* (discussed in Section 3.2 of this text), a term he coined. In the 1840s De Morgan made fundamental contributions to the development of symbolic logic. He invented notations that helped him prove propositional equivalences, such as the laws that are named after him. In 1842 De Morgan presented what was perhaps the first precise definition of a limit and developed some tests for convergence of infinite series. De Morgan was also interested in the history of mathematics and wrote biographies of Newton and Halley.

In 1837 De Morgan married Sophia Frend, who wrote his biography in 1882. De Morgan's research, writing, and teaching left little time for his family or social life. Nevertheless, he was noted for his kindness, humor, and wide range of knowledge.

TABLE 4 **A Demonstration That $p \vee (q \wedge r)$ and $(p \vee q) \wedge (p \vee r)$ Are Logically Equivalent.**

p	q	r	$q \wedge r$	$p \vee (q \wedge r)$	$p \vee q$	$p \vee r$	$(p \vee q) \wedge (p \vee r)$
T	T	T	T	T	T	T	T
T	T	F	F	T	T	T	T
T	F	T	F	T	T	T	T
T	F	F	F	T	T	T	T
F	T	T	T	T	T	T	T
F	T	F	F	F	T	F	F
F	F	T	F	F	F	T	F
F	F	F	F	F	F	F	F

Table 5 contains some important equivalences.* In these equivalences, **T** denotes any proposition that is always true and **F** denotes any proposition that is always false. The reader is asked to verify these equivalences in the exercises at the end of the section.

The associative law for disjunction shows that the expression $p \vee q \vee r$ is well defined, in the sense that it does not matter whether we first take the disjunction of p and

TABLE 5 **Logical Equivalences.**

Equivalence	*Name*
$p \wedge \mathbf{T} \Longleftrightarrow p$ $p \vee \mathbf{F} \Longleftrightarrow p$	Identity laws
$p \vee \mathbf{T} \Longleftrightarrow \mathbf{T}$ $p \wedge \mathbf{F} \Longleftrightarrow \mathbf{F}$	Domination laws
$p \vee p \Longleftrightarrow p$ $p \wedge p \Longleftrightarrow p$	Idempotent laws
$\neg(\neg p) \Longleftrightarrow p$	Double negation law
$p \vee q \Longleftrightarrow q \vee p$ $p \wedge q \Longleftrightarrow q \wedge p$	Commutative laws
$(p \vee q) \vee r \Longleftrightarrow p \vee (q \vee r)$ $(p \wedge q) \wedge r \Longleftrightarrow p \wedge (q \wedge r)$	Associative laws
$p \vee (q \wedge r) \Longleftrightarrow (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \Longleftrightarrow (p \wedge q) \vee (p \wedge r)$	Distributive laws
$\neg(p \wedge q) \Longleftrightarrow \neg p \vee \neg q$ $\neg(p \vee q) \Longleftrightarrow \neg p \wedge \neg q$	De Morgan's laws

---

*These identities are a special case of identities that hold for any Boolean algebra. Compare them with set identities in Table 1 in Section 1.5 and with Boolean identities in Table 5 in Section 9.1.

TABLE 6 Some Useful Logical Equivalences.

$p \vee \neg p \iff \mathbf{T}$
$p \wedge \neg p \iff \mathbf{F}$
$(p \rightarrow q) \iff (\neg p \vee q)$

q and then the disjunction of $p \vee q$ with r, or if we first take the disjunction of q and r and then take the disjunction of p and $q \vee r$. Similarly, the expression $p \wedge q \wedge r$ is well defined. By extending this reasoning, it follows that $p_1 \vee p_2 \vee \cdots \vee p_n$ and $p_1 \wedge p_2 \wedge \cdots \wedge p_n$ are well defined whenever $p_1, p_2, \ldots, p_n$ are propositions. Furthermore, note that De Morgan's laws extend to

$$\neg(p_1 \vee p_2 \cdots \vee p_n) \iff (\neg p_1 \wedge \neg p_2 \wedge \cdots \wedge \neg p_n)$$

and

$$\neg(p_1 \wedge p_2 \cdots \wedge p_n) \iff (\neg p_1 \vee \neg p_2 \vee \cdots \vee \neg p_n).$$

(Methods for proving these identities will be given in Chapter 3.)

The logical equivalences in Table 5, as well as any others that have been established (such as those shown in Table 6), can be used to construct additional logical equivalences. The reason for this is that a proposition in a compound proposition can be replaced by one that is logically equivalent to it without changing the truth value of the compound proposition. This technique is illustrated in Examples 5 and 6, where we also use the fact that if p and q are logically equivalent and q and r are logically equivalent, then p and r are logically equivalent (see Exercise 40).

EXAMPLE 5 Show that $\neg(p \vee (\neg p \wedge q))$ and $\neg p \wedge \neg q$ are logically equivalent.

Solution: We could use a truth table to show that these compound propositions are equivalent. Instead, we will establish this equivalence by developing a series of logical equivalences, using one of the equivalences in Table 5 at a time, starting with $\neg(p \vee (\neg p \wedge q))$ and ending with $\neg p \wedge \neg q$. We have the following equivalences.

$$\begin{aligned}
\neg(p \vee (\neg p \wedge q)) &\iff \neg p \wedge \neg(\neg p \wedge q) && \text{from the second De Morgan's law}\\
&\iff \neg p \wedge [\neg(\neg p) \vee \neg q] && \text{from the first De Morgan's law}\\
&\iff \neg p \wedge (p \vee \neg q) && \text{from the double negation law}\\
&\iff (\neg p \wedge p) \vee (\neg p \wedge \neg q) && \text{from the distributive law}\\
&\iff \mathbf{F} \vee (\neg p \wedge \neg q) && \text{since } \neg p \wedge p \iff \mathbf{F}\\
&\iff (\neg p \wedge \neg q) \vee \mathbf{F} && \text{from the law for disjunction}\\
&\iff \neg p \wedge \neg q && \text{from the identity law for } \mathbf{F}
\end{aligned}$$

Consequently $\neg(p \vee (\neg p \wedge q))$ and $\neg p \wedge \neg q$ are logically equivalent. ■

EXAMPLE 6 *Show that $(p \wedge q) \rightarrow (p \vee q)$ is a tautology.*

Solution: To show that this statement is a tautology, we will use logical equivalences to demonstrate that it is logically equivalent to **T**. (*Note:* This could also be done using a truth table.)

$$\begin{aligned}(p \wedge q) \rightarrow (p \vee q) &\iff \neg(p \wedge q) \vee (p \vee q) && \text{by Example 3}\\ &\iff (\neg p \vee \neg q) \vee (p \vee q) && \text{by the first De Morgan's law}\\ &\iff (\neg p \vee p) \vee (\neg q \vee q) && \text{by the associative and commutative laws for disjunction}\\ &\iff \mathbf{T} \vee \mathbf{T} && \text{by Example 1 and the commutative law for disjunction}\\ &\iff \mathbf{T} && \text{by the domination law} \quad \blacksquare\end{aligned}$$

A truth table can be used to determine whether a compound proposition is a tautology. This can be done by hand for a proposition with a small number of variables, but when the number of variables grows, this becomes impractical. For instance, there are $2^{20} = 1{,}048{,}576$ rows in the truth value table for a proposition with 20 variables. Clearly, you need a computer to help you determine, in this way, whether a compound proposition in 20 variables is a tautology. But when there are 1000 variables, can even a computer determine in a reasonable amount of time whether a compound proposition is a tautology? Checking every one of the 2^{1000} (a number with more than 300 decimal digits) possible combinations of truth values simply cannot be done by a computer in even trillions of years. Furthermore, no other procedures are known that a computer can follow to determine in a reasonable amount of time whether a compound proposition in such a large number of variables is a tautology. We will study questions such as this in Chapter 2, when we study the complexity of algorithms.

Exercises

1. Use truth tables to verify the following equivalences.
  a) $p \wedge \mathbf{T} \iff p$ **b)** $p \vee \mathbf{F} \iff p$
  c) $p \wedge \mathbf{F} \iff \mathbf{F}$ **d)** $p \vee \mathbf{T} \iff \mathbf{T}$
  e) $p \vee p \iff p$ **f)** $p \wedge p \iff p$

2. Show that $\neg(\neg p)$ and p are logically equivalent.

3. Use truth tables to verify the commutative laws
  a) $p \vee q \iff q \vee p$
  b) $p \wedge q \iff q \wedge p$

4. Use truth tables to verify the associative laws
  a) $(p \vee q) \vee r \iff p \vee (q \vee r)$
  b) $(p \wedge q) \wedge r \iff p \wedge (q \wedge r)$

5. Use truth tables to verify the distributive law $p \wedge (q \vee r) \iff (p \wedge q) \vee (p \wedge r)$.

6. Use a truth table to verify the equivalence $\neg(p \wedge q) \iff \neg p \vee \neg q$.

7. Show that each of the following implications is a tautology by using truth tables.
  a) $(p \wedge q) \rightarrow p$ **b)** $p \rightarrow (p \vee q)$
  c) $\neg p \rightarrow (p \rightarrow q)$ **d)** $(p \wedge q) \rightarrow (p \rightarrow q)$
  e) $\neg(p \rightarrow q) \rightarrow p$ **f)** $\neg(p \rightarrow q) \rightarrow \neg q$

8. Show that each of the following implications is a tautology by using truth tables.
  a) $[\neg p \wedge (p \vee q)] \rightarrow q$
  b) $[(p \rightarrow q) \wedge (q \rightarrow r)] \rightarrow (p \rightarrow r)$
  c) $[p \wedge (p \rightarrow q)] \rightarrow q$
  d) $[(p \vee q) \wedge (p \rightarrow r) \wedge (q \rightarrow r)] \rightarrow r$

Ada Augusta, Countess of Lovelace (1815–1852). Ada Augusta was the only child from the marriage of the famous poet Lord Byron and Annabella Millbanke, who separated when Ada was 1 month old. She was raised by her mother, who encouraged her intellectual talents. She was taught by the mathematicians William Frend and Augustus De Morgan. In 1838 she married Lord King, later elevated to Earl of Lovelace. Together they had three children.

Ada Augusta continued her mathematical studies after her marriage, assisting Charles Babbage in his work on an early computing machine, called the Analytic Engine. The most complete accounts of this machine are found in her writings. After 1845 she and Babbage worked toward the development of a system to predict horse races. Unfortunately, their system did not work well, leaving Ada heavily in debt at the time of her death. The programming language Ada is named in honor of the Countess of Lovelace.

9. Show that each implication in Exercise 7 is a tautology without using truth tables.
10. Show that each implication in Exercise 8 is a tautology without using truth tables.
11. Verify the following equivalences, which are known as the **absorption laws.**
    a) $[p \vee (p \wedge q)] \Longleftrightarrow p$
    b) $[p \wedge (p \vee q)] \Longleftrightarrow p$
12. Determine whether $(\neg p \wedge (p \rightarrow q)) \rightarrow \neg q$ is a tautology.
13. Determine whether $(\neg q \wedge (p \rightarrow q)) \rightarrow \neg p$ is a tautology.
14. Show that $p \leftrightarrow q$ and $(p \wedge q) \vee (\neg p \wedge \neg q)$ are equivalent.
15. Show that $(p \rightarrow q) \rightarrow r$ and $p \rightarrow (q \rightarrow r)$ are not equivalent.
16. Show that $p \rightarrow q$ and $\neg q \rightarrow \neg p$ are logically equivalent.
17. Show that $\neg p \leftrightarrow q$ and $p \leftrightarrow \neg q$ are logically equivalent.
18. Show that $\neg(p \oplus q)$ and $p \leftrightarrow q$ are logically equivalent.
19. Show that $\neg(p \leftrightarrow q)$ and $\neg p \leftrightarrow q$ are logically equivalent.

The **dual** of a compound proposition that contains only the logical operators $\vee$, $\wedge$, and $\neg$ is the proposition obtained by replacing each $\vee$ by $\wedge$, each $\wedge$ by $\vee$, each **T** by **F**, and each **F** by **T**. The dual of proposition s is denoted by s^*.

20. Find the dual of each of the following propositions.
    a) $p \wedge \neg q \wedge \neg r$
    b) $(p \wedge q \wedge r) \vee s$
    c) $(p \vee \mathbf{F}) \wedge (q \vee \mathbf{T})$
21. Show that $(s^*)^* = s$.
22. Show that the logical equivalences in Table 5, except for the double negation law, come in pairs, where each pair contains propositions that are duals of each other.
**23. Why are the duals of two equivalent compound propositions also equivalent, where these compound propositions contain only the operators $\wedge$, $\vee$, and $\neg$?
24. Find a compound proposition involving the propositions p, q, and r that is true when p and q are true and r is false, but is false otherwise. (*Hint:* Use a conjunction of each proposition or its negation.)
25. Find a compound proposition involving the propositions p, q, and r that is true when exactly two of p, q, and r are true and is false otherwise. (*Hint:* Form a disjunction of conjunctions. Include a conjunction for each combination of values for which the proposition is true. Each conjunction should include each of the three propositions or their negations.)
26. Suppose that a truth table in n propositional variables is specified. Show that a compound proposition with this truth table can be formed by taking the disjunction of conjunctions of the variables or their negations, with one conjunction included for each combination of values for which the compound proposition is true. The resulting compound proposition is said to be in **disjunctive normal form.**

A collection of logical operators is called **functionally complete** if every compound proposition is logically equivalent to a compound proposition involving only these logical operators.

27. Show that $\neg$, $\wedge$, and $\vee$ form a functionally complete collection of logical operators. (*Hint:* Use the fact that every proposition is logically equivalent to one in disjunctive normal form, as shown in Exercise 26.)
*28. Show that $\neg$ and $\wedge$ form a functionally complete collection of logical operators. (*Hint:* First use De Morgan's law to show that $p \vee q$ is equivalent to $\neg(\neg p \wedge \neg q)$.)
*29. Show that $\neg$ and $\vee$ form a functionally complete collection of logical operators.

The following exercises involve the logical operators *NAND* and *NOR*. The proposition p *NAND* q is true when either p or q, or both, are false; and it is false when both p and q are true. The proposition p *NOR* q is true when both p and q are false, and it is false otherwise. The propositions p *NAND* q and p *NOR* q are denoted by $p \mid q$ and $p \downarrow q$, respectively. (The operators $\mid$ and $\downarrow$ are called the **Sheffer stroke** and the **Peirce arrow** after H. M. Sheffer and C. S. Peirce, respectively.)

30. Construct a truth table for the logical operator *NAND*.
31. Show that $p \mid q$ is logically equivalent to $\neg(p \wedge q)$.
32. Construct a truth table for the logical operator *NOR*.
33. Show that $p \downarrow q$ is logically equivalent to $\neg(p \vee q)$.
34. In this exercise we will show that $\{\downarrow\}$ is a functionally complete collection of logical operators.
    a) Show that $p \downarrow p$ is logically equivalent to $\neg p$.
    b) Show that $(p \downarrow q) \downarrow (p \downarrow q)$ is logically equivalent to $p \vee q$.
    c) Conclude from parts (a) and (b), and Exercise 29, that $\{\downarrow\}$ is a functionally complete collection of logical operators.
*35. Find a proposition equivalent to $p \rightarrow q$ using only the logical operator $\downarrow$.
36. Show that $\{\mid\}$ is a functionally complete collection of logical operators.
37. Show that $p \mid q$ and $q \mid p$ are equivalent.
38. Show that $p \mid (q \mid r)$ and $(p \mid q) \mid r$ are not equivalent, so that the logical operator $\mid$ is not associative.
*39. How many different truth tables of compound propositions are there that involve the propositions p and q?
40. Show that if p, q, and r are compound propositions such that p and q are logically equivalent and q and r are logically equivalent, then p and r are logically equivalent.
41. The following sentence is taken from the specification of a telephone system: "If the directory data base is opened, then the monitor is put in a closed state, if the system is not in its initial state." This specification is hard to understand since it involves two implications. Find an equivalent, easier-to-understand specification that involves disjunctions and negations but not implications.

1.3

Predicates and Quantifiers

INTRODUCTION

Statements involving variables, such as

$$\text{“}x > 3\text{,”} \quad \text{“}x = y + 3\text{,”} \quad \text{and} \quad \text{“}x + y = z\text{,”}$$

are often found in mathematical assertions and in computer programs. These statements are neither true nor false when the values of the variables are not specified. In this section we will discuss the ways that propositions can be produced from such statements.

The statement "x is greater than 3" has two parts. The first part, the variable x, is the subject of the statement. The second part—the **predicate,** "is greater than 3"—refers to a property that the subject of the statement can have. We can denote the statement "x is greater than 3" by $P(x)$, where P denotes the predicate "is greater than 3" and x is the variable. The statement $P(x)$ is also said to be the value of the **propositional function** P at x. Once a value has been assigned to the variable x, the statement $P(x)$ becomes a proposition and has a truth value. Consider the following example.

EXAMPLE 1 Let $P(x)$ denote the statement "$x > 3$." What are the truth values of $P(4)$ and $P(2)$?

---

web

Charles Sanders Peirce (1839–1914). Many consider Charles Peirce the most original and versatile intellect from the United States; he was born in Cambridge, Massachusetts. His father, Benjamin Peirce, was a professor of mathematics and natural philosophy at Harvard. Peirce attended Harvard (1855–1859) and received a Harvard master of arts degree (1862) and an advanced degree in chemistry from the Lawrence Scientific School (1863). His father encouraged him to pursue a career in science, but instead he chose to study logic and scientific methodology.

In 1861, Peirce became an aide in the United States Coast Survey, with the goal of better understanding scientific methodology. His service for the Survey exempted him from military service during the Civil War. While working for the Survey, Peirce carried out astronomical and geodesic work. He made fundamental contributions to the design of pendulums and to map projections, applying new mathematical developments in the theory of elliptic functions. He was the first person to use the wavelength of light as a unit of measurement. Peirce rose to the position of Assistant for the Survey, a position he held until he was forced to resign in 1891 when he disagreed with the direction taken by the Survey's new administration.

Although making his living from work in the physical sciences, Peirce developed a hierarchy of sciences, with mathematics at the top rung, in which the methods of one science could be adapted for use by those sciences under it in the heirarchy. He was also the founder of the American philosophical theory of pragmatism.

The only academic position Peirce ever held was as a lecturer in logic at Johns Hopkins University in Baltimore from 1879 to 1884. His mathematical work during this time included contributions to logic, set theory, abstract algebra, and the philosophy of mathematics. His work is still relevant today; some of his work on logic has been recently applied to artificial intelligence. Peirce believed that the study of mathematics could develop the mind's powers of imagination, abstraction, and generalization. His diverse activities after retiring from the Survey included writing for newspapers and journals, contributing to scholarly dictionaries, translating scientific papers, guest lecturing, and textbook writing. Unfortunately, the income from these pursuits was insufficient to protect him and his second wife from abject poverty. He was supported in his later years by a fund created by his many admirers and administered by the philosopher William James, his lifelong friend. Although Peirce wrote and published voluminously in a vast range of subjects, he left more than 100,000 pages of unpublished manuscripts. Because of the difficulty of studying his unpublished writings, scholars have only recently started to understand some of his varied contributions. A group of people is devoted to making his work available over the Internet to bring a better appreciation of Peirce's accomplishments to the world.

Solution: The statement $P(4)$ is obtained by setting $x = 4$ in the statement "$x > 3$." Hence, $P(4)$, which is the statement "$4 > 3$," is true. However, $P(2)$, which is the statement "$2 > 3$," is false. ■

We can also have statements that involve more than one variable. For instance, consider the statement "$x = y + 3$." We can denote this statement by $Q(x, y)$, where x and y are variables and Q is the predicate. When values are assigned to the variables x and y, the statement $Q(x, y)$ has a truth value.

EXAMPLE 2 Let $Q(x, y)$ denote the statement "$x = y + 3$." What are the truth values of the propositions $Q(1, 2)$ and $Q(3, 0)$?

Solution: To obtain $Q(1, 2)$, set $x = 1$ and $y = 2$ in the statement $Q(x, y)$. Hence, $Q(1, 2)$ is the statement "$1 = 2 + 3$," which is false. The statement $Q(3, 0)$ is the proposition "$3 = 0 + 3$," which is true. ■

Similarly, we can let $R(x, y, z)$ denote the statement "$x + y = z$." When values are assigned to the variables x, y, and z, this statement has a truth value.

EXAMPLE 3 What are the truth values of the propositions $R(1, 2, 3)$ and $R(0, 0, 1)$?

Solution: The proposition $R(1, 2, 3)$ is obtained by setting $x = 1$, $y = 2$, and $z = 3$ in the statement $R(x, y, z)$. We see that $R(1, 2, 3)$ is the statement "$1 + 2 = 3$," which is true. Also note that $R(0, 0, 1)$, which is the statement "$0 + 0 = 1$," is false. ■

In general, a statement involving the n variables $x_1, x_2, \ldots, x_n$ can be denoted by

$P(x_1, x_2, \ldots, x_n)$.

A statement of the form $P(x_1, x_2, \ldots, x_n)$ is the value of the **propositional function** P at the n-tuple $(x_1, x_2, \ldots, x_n)$, and P is also called a *predicate*.

Propositional functions occur in computer programs, as the following example demonstrates.

EXAMPLE 4 Consider the statement

if $x > 0$ **then** $x := x + 1$.

When this statement is encountered in a program, the value of the variable x at that point in the execution of the program is inserted into $P(x)$, which is "$x > 0$." If $P(x)$ is true for this value of x, the assignment statement $x := x + 1$ is executed, so the value of x is increased by 1. If $P(x)$ is false for this value of x, the assignment statement is not executed, so the value of x is not changed. ■

QUANTIFIERS

When all the variables in a propositional function are assigned values, the resulting statement has a truth value. However, there is another important way, called **quantification,** to create a proposition from a propositional function. Two types of quantification will be discussed here, namely, universal quantification and existential quantification.

Many mathematical statements assert that a property is true for all values of a variable in a particular domain, called the **universe of discourse.** Such a statement is expressed using a universal quantification. The universal quantification of a propositional function is the proposition that asserts that $P(x)$ is true for all values of x in the universe of discourse. The universe of discourse specifies the possible values of the variable x.

DEFINITION 1. The ***universal quantification*** **of $P(x)$ is the proposition**

"$P(x)$ is true for all values of x in the universe of discourse."

The notation

$$\forall x\, P(x)$$

denotes the universal quantification of $P(x)$. Here $\forall$ is called the **universal quantifier.** The proposition $\forall x\, P(x)$ is also expressed as

"for all x $P(x)$" or "for every x $P(x)$."

Remark: It is best to avoid the word "any" since it is often ambiguous as to whether it means "every" or "some." In some cases, "any" is unambiguous, such as when it is used in negatives, for example, "there is not any reason not to study hard."

EXAMPLE 5 Express the statement

"Every student in this class has studied calculus"

as a universal quantification.

Solution: Let $P(x)$ denote the statement

"x has studied calculus."

Then the statement "Every student in this class has studied calculus" can be written as $\forall x\, P(x)$, where the universe of discourse consists of the students in this class.

This statement can also be expressed as

$$\forall x\,(S(x) \rightarrow P(x))$$

where $S(x)$ is the statement

"x is in this class."

$P(x)$ is as before, and the universe of discourse is the set of all students. ■

Example 5 illustrates that there is often more than one good way to express a quantification.

EXAMPLE 6 Let $P(x)$ be the statement "$x + 1 > x$." What is the truth value of the quantification $\forall x\, P(x)$, where the universe of discourse is the set of real numbers?

Solution: Since $P(x)$ is true for all real numbers x, the quantification

$$\forall x\, P(x)$$

is true. ■

EXAMPLE 7 Let $Q(x)$ be the statement "$x < 2$." What is the truth value of the quantification $\forall x\, Q(x)$, where the universe of discourse is the set of real numbers?

Solution: $Q(x)$ is not true for all real numbers x, since, for instance, $Q(3)$ is false. Thus

$$\forall x\, Q(x)$$

is false. ■

When all of the elements in the universe of discourse can be listed—say, $x_1, x_2, \ldots, x_n$—it follows that the universal quantification $\forall x\, P(x)$ is the same as the conjunction

$$P(x_1) \wedge P(x_2) \wedge \cdots \wedge P(x_n),$$

since this conjunction is true if and only if $P(x_1), P(x_2), \ldots, P(x_n)$ are all true.

EXAMPLE 8 What is the truth value of $\forall x\, P(x)$, where $P(x)$ is the statement "$x^2 < 10$" and the universe of discourse consists of the positive integers not exceeding 4?

Solution: The statement $\forall x\, P(x)$ is the same as the conjunction

$$P(1) \wedge P(2) \wedge P(3) \wedge P(4),$$

since the universe of discourse consists of the integers 1, 2, 3, and 4. Since $P(4)$, which is the statement "$4^2 < 10$," is false, it follows that $\forall x\, P(x)$ is false. ■

Many mathematical statements assert that there is an element with a certain property. Such statements are expressed using existential quantification. With existential quantification, we form a proposition that is true if and only if $P(x)$ is true for at least one value of x in the universe of discourse.

DEFINITION 2. The *existential quantification* of $P(x)$ is the proposition

"There exists an element x in the universe of discourse such that $P(x)$ is true."

We use the notation

$$\exists x\, P(x)$$

for the existential quantification of $P(x)$. Here $\exists$ is called the **existential quantifier.** The existential quantification $\exists x\, P(x)$ is also expressed as

"There is an x such that $P(x)$,"
"There is at least one x such that $P(x)$,"

or

"For some x $P(x)$."

EXAMPLE 9 Let $P(x)$ denote the statement "$x > 3$." What is the truth value of the quantification $\exists x\, P(x)$, where the universe of discourse is the set of real numbers?

Solution: Since "$x > 3$" is true—for instance, when $x = 4$—the existential quantification of $P(x)$, which is $\exists x\, P(x)$, is true. ■

EXAMPLE 10 Let $Q(x)$ denote the statement "$x = x + 1$." What is the truth value of the quantification $\exists x\, Q(x)$, where the universe of discourse is the set of real numbers?

Solution: Since $Q(x)$ is false for every real number x, the existential quantification of $Q(x)$, which is $\exists x\, Q(x)$, is false. ■

When all of the elements in the universe of discourse can be listed—say, $x_1, x_2, \ldots, x_n$—the existential quantification $\exists x\, P(x)$ is the same as the disjunction

$$P(x_1) \vee P(x_2) \vee \cdots \vee P(x_n),$$

since this disjunction is true if and only if at least one of $P(x_1), P(x_2), \ldots, P(x_n)$ is true.

EXAMPLE 11 What is the truth value of $\exists x\, P(x)$ where $P(x)$ is the statement "$x^2 > 10$" and the universe of discourse consists of the positive integers not exceeding 4?

Solution: Since the universe of discourse is $\{1, 2, 3, 4\}$, the proposition $\exists x\, P(x)$ is the same as the disjunction

$$P(1) \vee P(2) \vee P(3) \vee P(4).$$

Since $P(4)$, which is the statement "$4^2 > 10$," is true, it follows that $\exists x\, P(x)$ is true. ■

Table 1 summarizes the meaning of the universal and the existential quantifiers.

It is sometimes helpful to think in terms of looping and searching when determining the truth value of a quantification. Suppose that there are n objects in the universe of discourse for the variable x. To determine whether $\forall x\, P(x)$ is true, we can loop through all n values of x to see if $P(x)$ is always true. If we encounter a value x for which $P(x)$ is false, then we have shown that $\forall x\, P(x)$ is false. Otherwise, $\forall x\, P(x)$ is true. To see whether $\exists x\, P(x)$ is true, we loop through the n values of x searching for a value for

TABLE 1 Quantifiers.

Statement	*When True?*	*When False?*
$\forall x\, P(x)$	$P(x)$ is true for every x.	There is an x for which $P(x)$ is false.
$\exists x\, P(x)$	There is an x for which $P(x)$ is true.	$P(x)$ is false for every x.

which $P(x)$ is true. If we find one, then $\exists x\, P(x)$ is true. If we never find such an x, we have determined that $\exists x\, P(x)$ is false. (Note that this searching procedure does not apply if there are infinitely many values in the universe of discourse. However, it is still a useful way of thinking about the truth values of quantifications.)

Sometimes expressions involving quantifiers can be quite complicated. Translating a complicated expression into English helps understanding of its meaning. The first step in this translation is to write out what each quantifier means. The next step is to express this meaning in a simpler sentence. Consider the following examples.

EXAMPLE 12 Translate the statement

$$\forall x(C(x) \vee \exists y(C(y) \wedge F(x,y)))$$

into English, where $C(x)$ is "x has a computer," $F(x,y)$ is "x and y are friends," and the universe of discourse for both x and y is the set of all students in your school.

Solution: The statement says that for every student x in your school x has a computer or there is a student y such that y has a computer and x and y are friends. In other words, every student in your school has a computer or has a friend who has a computer. ■

EXAMPLE 13 Translate the statement

$$\exists x\forall y\forall z((F(x,y) \wedge F(x,z) \wedge (y \neq z)) \rightarrow \neg F(y,z))$$

into English, where $F(a,b)$ means a and b are friends and the universe of discourse for x, y, and z is the set of all students in your school.

Solution. This statement says that there is a student x such that for all students y and all students z other than y, if x and y are friends and x and z are friends, then y and z are not friends. In other words, there is a student none of whose friends are also friends with each other. ■

Complicated expressions involving quantifiers also arise in mathematical statements. This is illustrated in the following example.

EXAMPLE 14 Assume that the universe of discourse for the variables x and y is the set of all real numbers. The statement

$$\forall x\forall y(x + y = y + x)$$

says that $x + y = y + x$ for all real numbers x and y. This is the commutative law for addition of real numbers. Likewise, the statement

$$\forall x\exists y(x + y = 0)$$

says that for every real number x there is a real number y such that $x + y = 0$. This states that every real number has an additive inverse. Similarly, the statement

$$\forall x\forall y\forall z(x + (y + z) = (x + y) + z)$$

is the associative law for addition of real numbers. ■

TRANSLATING SENTENCES INTO LOGICAL EXPRESSIONS

In Section 1.1 we illustrated the process of translating English sentences into logical expressions involving propositions and logical connectives. Now that we have discussed quantifiers, we can express a wider variety of English sentences using logical expressions. Doing so eliminates ambiguity and makes it possible to reason with these sentences. (Section 3.1 covers rules of inference for reasoning with logical expressions.)

The following examples show how to use logical operators and quantifiers to express English sentences, similar to the kind that occur frequently in mathematical statements, in logic programming, and in artificial intelligence.

EXAMPLE 15 Express the statements "Some student in this class has visited Mexico" and "Every student in this class has visited either Canada or Mexico" using quantifiers.

Solution: Let the universe of discourse for the variable x be the set of students in your class. Let $M(x)$ be the statement "x has visited Mexico" and $C(x)$ the statement "x has visited Canada." The statement "Some student in this class has visited Mexico" can be written as $\exists x M(x)$. The statement "Every student in this class has visited either Canada or Mexico" can be written as $\forall x(C(x) \vee M(x))$ (assuming that the inclusive, rather than the exclusive, or is what is meant here). ■

EXAMPLE 16 Express the statement "Everyone has exactly one best friend" as a logical expression.

Solution: Let $B(x, y)$ be the statement "y is the best friend of x." To translate the sentence in the example, note that it says that for every person x there is another person y such that y is the best friend of x and that if z is a person other than y, then z is not the best friend of x. Consequently, we can translate the sentence as

$$\forall x \exists y \forall z (B(x, y) \wedge ((z \neq y) \rightarrow \neg B(x, z))).$$ ■

EXAMPLE 17 Express the statement "If somebody is female and is a parent, then this person is someone's mother" as a logical expression.

Solution: Let $F(x)$ be the statement "x is female," let $P(x)$ be the statement "x is a parent," and let $M(x, y)$ be the statement "x is the mother of y." Since the statement in the example pertains to all people, we can write it symbolically as

$$\forall x((F(x) \wedge P(x)) \rightarrow \exists y M(x, y)).$$ ■

EXAMPLE 18 Use quantifiers to express the statement "There is a woman who has taken a flight on every airline in the world."

Solution: Let $P(w, f)$ be "w has taken f" and $Q(f, a)$ be "f is a flight on a." We can express the statement as

$$\exists w \forall a \exists f (P(w, f) \wedge Q(f, a)),$$

where the universes of discourse for w, f, and a consist of all the women in the world, all airplane flights, and all airlines, respectively.

The statement could also be expressed as

$$\exists w \forall a \exists f\, R(w, f, a),$$

where $R(w, f, a)$ is "w has taken f on a." Although this is more compact, it somewhat obscures the relationships between the variables. Consequently, the first solution is usually preferable. ■

As mentioned earlier, quantifiers are often used in the definition of mathematical concepts. One example that you may be familiar with is the concept of limit, which is important in calculus.

EXAMPLE 19 (Calculus required) Express the definition of a limit using quantifiers.

Solution: Recall that the definition of the statement

$$\lim_{x \to a} f(x) = L$$

is: For every real number $\epsilon > 0$ there exists a real number $\delta > 0$ such that $|f(x) - L| < \epsilon$ whenever $0 < |x - a| < \delta$. This definition of a limit can be phrased in terms of quantifiers by

$$\forall \epsilon \exists \delta \forall x (0 < |x - a| < \delta \rightarrow |f(x) - L| < \epsilon),$$

where the universe of discourse for the variables δ and ϵ is the set of positive real numbers and for x is the set of real numbers.

This definition can also be expressed as

$$\forall \epsilon > 0 \exists \delta > 0 \forall x (0 < |x - a| < \delta \rightarrow |f(x) - L| < \epsilon)$$

when the universe of discourse for the variables ϵ and δ is the set of all real numbers, rather than the set of positive real numbers. ■

EXAMPLES FROM LEWIS CARROLL (optional)

Lewis Carroll (really C. L. Dodgson writing under a pseudonym), the author of *Alice in Wonderland*, is also the author of several works on symbolic logic. His books contain many examples of reasoning using quantifiers. The next two examples come from his book *Symbolic Logic;* other examples from that book are given in the exercise set at the end of this section. These examples illustrate how quantifiers are used to express various types of statements.

Charles Lutwidge Dodgson (1832–1898). We know Charles Dodgson as *Lewis Carroll*—the pseudonym he used in his writings on logic. Dodgson, the son of a clergyman, was the third of 11 children, all of whom stuttered. He was uncomfortable in the company of adults and is said to have spoken without stuttering only to young girls, many of whom he entertained, corresponded with, and photographed (often in the nude). Although attracted to young girls, he was extremely puritanical and religious. His friendship with the three young daughters of Dean Liddell led to his writing *Alice in Wonderland*, which brought him money and fame.

Dodgson graduated from Oxford in 1854 and obtained his master of arts degree in 1857. He was appointed lecturer in mathematics at Christ Church College, Oxford, in 1855. He was ordained in the Church of England in 1861 but never practiced his ministry. His writings include articles and books on geometry, determinants, and the mathematics of tournaments and elections. (He also used the pseudonym Lewis Carroll for his many works on recreational logic.)

EXAMPLE 20 Consider the following statements. The first two are called *premises* and the third is called the *conclusion*. The entire set is called an *argument*.

"All lions are fierce."
"Some lions do not drink coffee."
"Some fierce creatures do not drink coffee."

(In Section 3.1 we will discuss the issue of determining whether the conclusion is a valid consequence of the premises. In this example, it is.) Let $P(x)$, $Q(x)$, and $R(x)$ be the statements "x is a lion," "x is fierce," and "x drinks coffee," respectively. Assuming that the universe of discourse is the set of all creatures, express the statements in the argument using quantifiers and $P(x)$, $Q(x)$, and $R(x)$.

Solution: We can express these statements as:

$$\forall x(P(x) \rightarrow Q(x)).$$
$$\exists x(P(x) \wedge \neg R(x)).$$
$$\exists x(Q(x) \wedge \neg R(x)).$$

Notice that the second statement cannot be written as $\exists x(P(x) \rightarrow \neg R(x))$. The reason is that $P(x) \rightarrow \neg R(x)$ is true whenever x is not a lion, so that $\exists x(P(x) \rightarrow \neg R(x))$ is true as long as there is at least one creature that is not a lion, even if every lion drinks coffee. Similarly, the third statement cannot be written as

$$\exists x(Q(x) \rightarrow \neg R(x)).$$ ■

EXAMPLE 21 Consider the following statements, of which the first three are premises and the fourth is a valid conclusion.

"All hummingbirds are richly colored."
"No large birds live on honey."
"Birds that do not live on honey are dull in color."
"Hummingbirds are small."

Let $P(x)$, $Q(x)$, $R(x)$, and $S(x)$ be the statements "x is a hummingbird," "x is large," "x lives on honey," and "x is richly colored," respectively. Assuming that the universe of discourse is the set of all birds, express the statements in the argument using quantifiers and $P(x)$, $Q(x)$, $R(x)$, and $S(x)$.

Solution: We can express the statements in the argument as:

$$\forall x(P(x) \rightarrow S(x)).$$
$$\neg\exists x(Q(x) \wedge R(x)).$$
$$\forall x(\neg R(x) \rightarrow \neg S(x)).$$
$$\forall x(P(x) \rightarrow \neg Q(x)).$$

(Note we have assumed that "small" is the same as "not large" and that "dull in color" is the same as "not richly colored." To show that the fourth statement is a valid conclusion of the first three, we need to use rules of inference that will be discussed in Section 3.1.) ■

BINDING VARIABLES

When a quantifier is used on the variable x or when we assign a value to this variable, we say that this occurrence of the variable is **bound**. An occurrence of a variable that is not bound by a quantifier or set equal to a particular value is said to be **free.** All the variables that occur in a propositional function must be bound to turn it into a proposition. This can be done using a combination of universal quantifiers, existential quantifiers, and value assignments.

Many mathematical statements involve multiple quantifications of propositional functions involving more than one variable. It is important to note that the order of the quantifiers is important, unless all the quantifiers are universal quantifiers or all are existential quantifiers. These remarks are illustrated by Examples 22, 23, and 24. In each of these examples the universe of discourse for each variable is the set of real numbers.

EXAMPLE 22 Let $P(x, y)$ be the statement "$x + y = y + x$." What is the truth value of the quantification $\forall x \forall y\, P(x, y)$?

Solution: The quantification

$$\forall x \forall y\, P(x, y)$$

denotes the proposition

"For all real numbers x and for all real numbers y, it is true that $x + y = y + x$."

Since $P(x, y)$ is true for all real numbers x and y, the proposition $\forall x \forall y\, P(x, y)$ is true. ■

EXAMPLE 23 Let $Q(x, y)$ denote "$x + y = 0$." What are the truth values of the quantifications $\exists y \forall x\, Q(x, y)$ and $\forall x \exists y\, Q(x, y)$?

Solution: The quantification

$$\exists y \forall x\, Q(x, y)$$

denotes the proposition

"There is a real number y such that for every real number x, $Q(x, y)$ is true."

No matter what value of y is chosen, there is only one value of x for which $x + y = 0$. Since there is no real number y such that $x + y = 0$ for all real numbers x, the statement $\exists y \forall x\, Q(x, y)$ is false.

The quantification

$$\forall x \exists y\, Q(x, y)$$

denotes the proposition

"For every real number x there is a real number y such that $Q(x, y)$ is true."

Given a real number x, there is a real number y such that $x + y = 0$; namely, $y = -x$. Hence, the statement $\forall x \exists y\, Q(x, y)$ is true. ■

Example 23 illustrates that the order in which quantifiers appear makes a difference. The statement $\exists y \forall x\, P(x, y)$ and $\forall x \exists y\, P(x, y)$ are not logically equivalent. The statement $\exists y \forall x\, P(x, y)$ is true if and only if there is a y that makes $P(x, y)$ true for every x. So, for this statement to be true, there must be a particular value of y for which $P(x, y)$ is true regardless of the choice of x. On the other hand, $\forall x\, \exists y\, P(x, y)$ is true if and only if for every value of x there is a value of y for which $P(x, y)$ is true. So, for this statement to be true, no matter which x you choose, there must be a value of y (possibly depending on the x you choose) for which $P(x, y)$ is true. In other words, in the second case y can depend on x, whereas in the first case y is a constant independent of x.

From these observations, it follows that if $\exists y\, \forall x\, P(x, y)$ is true, then $\forall x\, \exists y\, P(x, y)$ must also be true. However, if $\forall x\, \exists y\, P(x, y)$ is true, it is not necessary for $\exists y\, \forall x\, P(x, y)$ to be true. (See Supplementary Exercises 8 and 10 at the end of this chapter.)

In working with quantifications of more than one variable, it is sometimes helpful to think in terms of nested loops. (Of course, if there are infinitely many elements in the universe of discourse of some variable, we cannot actually loop through all values. Nevertheless, this way of thinking is helpful in understanding nested quantifiers.) For example, to see whether $\forall x\, \forall y\, P(x, y)$ is true, we loop through the values for x, and for each x we loop through the values for y. If we find that $P(x, y)$ is true for all values for x and y, we have determined that $\forall x\, \forall y\, P(x, y)$ is true. If we ever hit a value x for which we hit a value y for which $P(x, y)$ is false, we have shown that $\forall x\, \forall y\, P(x, y)$ is false.

Similarly, to determine whether $\forall x\, \exists y\, P(x, y)$ is true, we loop through the values for x. For each x we loop through the values for y until we find a y for which $P(x, y)$ is true. If for all x we hit such a y, then $\forall x\, \exists y\, P(x, y)$ is true; if for some x we never hit such a y, then $\forall x\, \exists y\, P(x, y)$ is false.

To see whether $\exists x\, \forall y\, P(x, y)$ is true, we loop through the values for x until we find an x for which $P(x, y)$ is always true when we loop through all values for y. Once we find such an x, we know that $\exists x\, \forall y\, P(x, y)$ is true. If we never hit such an x, then we know that $\exists x\, \forall y\, P(x, y)$ is false.

Finally, to see whether $\exists x\, \exists y\, P(x, y)$ is true, we loop through the values for x, where for each x we loop through the values for y until we hit an x for which we hit a y for which $P(x, y)$ is true. The statement $\exists x\, \exists y\, P(x, y)$ is false only if we never hit an x for which we hit a y such that $P(x, y)$ is true.

Table 2 summarizes the meanings of the different possible quantifications involving two variables.

Quantifications of more than two variables are also common, as Example 24 illustrates.

TABLE 2 **Quantifications of Two Variables.**

Statement	*When True?*	*When False?*
$\forall x\, \forall y\, P(x, y)$ $\forall y\, \forall x\, P(x, y)$	$P(x, y)$ is true for every pair x, y.	There is a pair x, y for which $P(x, y)$ is false.
$\forall x\, \exists y\, P(x, y)$	For every x there is a y for which $P(x, y)$ is true.	There is an x such that $P(x, y)$ is false for every y.
$\exists x\, \forall y\, P(x, y)$	There is an x for which $P(x, y)$ is true for every y.	For every x there is a y for which $P(x, y)$ is false.
$\exists x\, \exists y\, P(x, y)$ $\exists y\, \exists x\, P(x, y)$	There is a pair x, y for which $P(x, y)$ is true.	$P(x, y)$ is false for every pair x, y.

EXAMPLE 24 Let $Q(x, y, z)$ be the statement "$x + y = z$." What are the truth values of the statements $\forall x \forall y \exists z\, Q(x, y, z)$ and $\exists z \forall x \forall y\, Q(x, y, z)$?

Solution: Suppose that x and y are assigned values. Then, there exists a real number z such that $x + y = z$. Consequently, the quantification

$$\forall x \forall y \exists z\, Q(x, y, z),$$

which is the statement

> "For all real numbers x and for all real numbers y there is a real number z such that $x + y = z$,"

is true. The order of the quantification here is important, since the quantification

$$\exists z \forall x \forall y\, Q(x, y, z),$$

which is the statement

> "There is a real number z such that for all real numbers x and for all real numbers y it is true that $x + y = z$,"

is false, since there is no value of z that satisfies the equation $x + y = z$ for all values of x and y. ■

NEGATIONS

We will often want to consider the negation of a quantified expression. For instance, consider the negation of the statement

> "Every student in the class has taken a course in calculus."

This statement is a universal quantification, namely,

$$\forall x\, P(x),$$

where $P(x)$ is the statement "x has taken a course in calculus." The negation of this statement is "It is not the case that every student in the class has taken a course in calculus." This is equivalent to "There is a student in the class who has not taken a course in calculus." And this is simply the existential quantification of the negation of the original propositional function, namely,

$$\exists x\, \neg P(x).$$

This example illustrates the following equivalence:

$$\neg \forall x\, P(x) \Longleftrightarrow \exists x\, \neg P(x).$$

Suppose we wish to negate an existential quantification. For instance, consider the proposition "There is a student in this class who has taken a course in calculus." This is the existential quantification

$$\exists x\, Q(x),$$

where $Q(x)$ is the statement "x has taken a course in calculus." The negation of this statement is the proposition "It is not the case that there is a student in this class who has taken a course in calculus." This is equivalent to "Every student in this class has not taken calculus," which is just the universal quantification of the negation of the

TABLE 3 Negating Quantifiers.

Negation	*Equivalent Statement*	*When Is Negation True?*	*When False?*
$\neg\exists x\, P(x)$	$\forall x \neg P(x)$	$P(x)$ is false for every x.	There is an x for which $P(x)$ is true.
$\neg\forall x\, P(x)$	$\exists x \neg P(x)$	There is an x for which $P(x)$ is false.	$P(x)$ is true for every x.

original propositional function, or, phrased in the language of quantifiers,

$$\forall x\, \neg Q(x).$$

This example illustrates the equivalence

$$\neg\exists x\, Q(x) \Longleftrightarrow \forall x\, \neg Q(x).$$

Negations of quantifiers are summarized in Table 3.

Exercises

1. Let $P(x)$ denote the statement "$x \leq 4$." What are the truth values of the following?
- **a)** $P(0)$
- **b)** $P(4)$
- **c)** $P(6)$

2. Let $P(x)$ be the statement "the word x contains the letter a." What are the truth values of the following?
- **a)** P(orange)
- **b)** P(lemon)
- **c)** P(true)
- **d)** P(false)

3. Let $Q(x, y)$ denote the statement "x is the capital of y." What are the truth values of the following?
- **a)** Q(Denver, Colorado)
- **b)** Q(Detroit, Michigan)
- **c)** Q(Massachusetts, Boston)
- **d)** Q(New York, New York)

4. State the value of x after the statement **if** $P(x)$ **then** $x := 1$ is executed, where $P(x)$ is the statement "$x > 1$," if the value of x when this statement is reached is
- **a)** $x = 0$
- **b)** $x = 1$
- **c)** $x = 2$

5. Let $P(x)$ be the statement "x spends more than five hours every weekday in class," where the universe of discourse for x is the set of students. Express each of the following quantifications in English.
- **a)** $\exists x\, P(x)$
- **b)** $\forall x\, P(x)$
- **c)** $\exists x\, \neg P(x)$
- **d)** $\forall x\, \neg P(x)$

6. Let $P(x, y)$ be the statement "x has taken class y," where the universe of discourse for x is the set of all students in your class and for y is the set of all computer science courses at your school. Express each of the following quantifications in English.
- **a)** $\exists x \exists y\, P(x, y)$
- **b)** $\exists x \forall y\, P(x, y)$
- **c)** $\forall x \exists y\, P(x, y)$
- **d)** $\exists y \forall x\, P(x, y)$
- **e)** $\forall y \exists x\, P(x, y)$
- **f)** $\forall x \forall y\, P(x, y)$

7. Let $W(x, y)$ mean that x has visited y, where the universe of discourse for x is the set of all students in your school and the universe of discourse for y is the set of all Web sites. Express each of the following statements by a simple English sentence.
- **a)** W(Sarah Smith, www.att.com)
- **b)** $\exists x\, W(x, \text{www.imdb.org})$
- **c)** $\exists y\, W(\text{Jose Orez}, y)$
- **d)** $\exists y\, (W(\text{Ashok Puri}, y) \wedge W(\text{Cindy Yoon}, y))$
- **e)** $\exists y \forall z (y \neq (\text{David Belcher}) \wedge (W(\text{David Belcher}, z) \rightarrow W(y, z)))$
- **f)** $\exists x \exists y \forall z ((x \neq y) \wedge (W(x, z) \leftrightarrow W(y, z)))$

8. Let $C(x, y)$ mean that x is enrolled in y, where the universe of discourse for x is the set of all students in your school and the universe of discourse for y is the set of all classes being given at your school. Express each of the following statements by a simple English sentence.
- **a)** C(Randy Goldberg, CS 252)
- **b)** $\exists x\, C(x, \text{Math } 695)$
- **c)** $\exists y\, C(\text{Carol Sitea}, y)$
- **d)** $\exists x\, (C(x, \text{Math } 222) \wedge C(x, \text{CS } 252))$
- **e)** $\exists x \exists y \forall z ((x \neq y) \wedge (C(x, z) \rightarrow C(y, z)))$
- **f)** $\exists x \exists y \forall z ((x \neq y) \wedge (C(x, z) \leftrightarrow C(y, z)))$

9. Let $P(x)$ be the statement "x can speak Russian" and let $Q(x)$ be the statement "x knows the computer language C++." Express each of the following sentences in terms of $P(x)$, $Q(x)$, quantifiers, and logical connectives. For the universe of discourse for quantifiers use the set of all students at your school.
- **a)** There is a student at your school who can speak Russian and who knows C++.

b) There is a student at your school who can speak Russian but who doesn't know C++.

c) Every student at your school either can speak Russian or knows C++.

d) No student at your school can speak Russian or knows C++.

10. Let $Q(x, y)$ be the statement "x has been a contestant on y." Express each of the following sentences in terms of $Q(x, y)$, quantifiers, and logical connectives, where the universe of discourse for x is the set of all students at your school and for y is the set of all quiz shows on television.

a) There is a student at your school who has been a contestant on a television quiz show.

b) No student at your school has ever been a contestant on a television quiz show.

c) There is a student at your school who has been a contestant on *Jeopardy* and on *Wheel of Fortune*.

d) Every television quiz show has had a student from your school as a contestant.

e) At least two students from your school have been contestants on *Jeopardy*.

11. Let $L(x, y)$ be the statement "x loves y," where the universe of discourse for both x and y is the set of all people in the world. Use quantifiers to express each of the following statements:

a) Everybody loves Jerry.

b) Everybody loves somebody.

c) There is somebody whom everybody loves.

d) Nobody loves everybody.

e) There is somebody whom Lydia does not love.

f) There is somebody whom no one loves.

g) There is exactly one person whom everybody loves.

h) There are exactly two people whom Lynn loves.

i) Everyone loves himself or herself.

j) There is someone who loves no one besides himself or herself.

12. Let $F(x, y)$ be the statement "x can fool y," where the universe of discourse is the set of all people in the world. Use quantifiers to express each of the following statements:

a) Everybody can fool Fred.

b) Evelyn can fool everybody.

c) Everybody can fool somebody.

d) There is no one who can fool everybody.

e) Everyone can be fooled by somebody.

f) No one can fool both Fred and Jerry.

g) Nancy can fool exactly two people.

h) There is exactly one person whom everybody can fool.

i) No one can fool himself or herself.

j) There is someone who can fool exactly one person besides himself or herself.

13. Let $S(x)$ be the predicate "x is a student," $F(x)$ the predicate "x is a faculty member," and $A(x, y)$ the predicate "x has asked y a question," where the universe of discourse is the set of all people associated with your school. Use quantifiers to express each of the following statements.

a) Lois has asked Professor Michaels a question.

b) Every student has asked Professor Gross a question.

c) Every faculty member has either asked Professor Miller a question or been asked a question by Professor Miller.

d) Some student has not asked any faculty member a question.

e) There is a faculty member who has never been asked a question by a student.

f) Some student has asked every faculty member a question.

g) There is a faculty member who has asked every other faculty member a question.

h) Some student has never been asked a question by a faculty member.

14. Let $I(x)$ be the statement "x has an Internet connection" and $C(x, y)$ be the statement "x and y have chatted over the Internet," where the universe of discourse for the variables x and y is the set of all students in your class. Use quantifiers to express each of the following statements.

a) Jerry does not have an Internet connection.

b) Rachel has not chatted over the Internet with Chelsea.

c) Jan and Sharon have never chatted over the Internet.

d) No one in the class has chatted with Bob.

e) Sanjay has chatted with everyone except Joseph.

f) Someone in your class does not have an Internet connection.

g) Not everyone in your class has an Internet connection.

h) Exactly one student in your class has an Internet connection.

i) Everyone except one student in your class has an Internet connection.

j) Everyone in your class with an Internet connection has chatted over the Internet with at least one other student in your class.

k) Someone in your class has an Internet connection but has not chatted with anyone else in your class.

l) There are two students in your class who have not chatted with over the Internet.

m) There is a student in your class who has chatted with everyone in your class over the Internet.

n) There are at least two students in your class who have not chatted with the same person in your class.

o) There are two students in the class who between them have chatted with everyone else in the class.

15. Let $M(x, y)$ be "x has sent y an e-mail message" and $T(x, y)$ be "x has telephoned y," where the universe of discourse is the set of all students in your class. Use quantifiers to express each of the following statements.

(*Assume that all e-mail messages that were sent are received, which is not the way things often work.*)

- **a)** Chou has never sent an e-mail message to Koko.
- **b)** Arlene has never sent an e-mail message to or telephoned Sarah.
- **c)** Jose has never received an e-mail message from Deborah.
- **d)** Every student in your class has sent e-mail message to Ken.
- **e)** No one in your class has telephoned Nina.
- **f)** Everyone in class has either telephoned Avi or sent him an e-mail message.
- **g)** There is a student in your class who has sent everyone else in your class an e-mail message.
- **h)** There is someone in your class who has either sent an e-mail message or telephoned everyone else in your class.
- **i)** There are two students in your class who have sent each other e-mail messages.
- **j)** There is a student who has sent himself or herself an e-mail message.
- **k)** There is a student in your class who has not received an e-mail message from anyone else in the class and who has not been called by any other student in the class.
- **l)** Every student in the class has either received an e-mail message or received a telephone call from another student in the class.
- **m)** There are at least two students in your class such that one student has sent the other e-mail and the second student has telephoned the first student.
- **n)** There are two students in your class who between them have sent an e-mail message to or telephoned everyone else in the class.

16. Use quantifiers to express each of the following statements.

- **a)** There is a student in this class who can speak Hindi.
- **b)** Every student in this class knows how to drive a car.
- **c)** Some student in this class has visited Alaska but has not visited Hawaii.
- **d)** All students in this class have learned at least one programming language.
- **e)** There is a student in this class who has taken every course offered by one of the departments in this school.
- **f)** Some student in this class grew up in the same town as exactly one other student in this class.
- **g)** Every student in this class has chatted with at least one other student in at least one on-line chat group.

17. Use quantifiers to express the following statements.

- **a)** Every computer science student needs a course in discrete mathematics.
- **b)** There is a student in this class who owns a personal computer.
- **c)** Every student in this class has taken at least one computer science course.
- **d)** There is a student in this class who has taken at least one course in computer science.
- **e)** Every student in this class has been in every building on campus.
- **f)** There is a student in this class who has been in every room of at least one building on campus.
- **g)** Every student in this class has been in at least one room of every building on campus.

18. A discrete mathematics class contains 1 mathematics major who is a freshman, 12 mathematics majors who are sophomores, 15 computer science majors who are sophomores, 2 mathematics majors who are juniors, 2 computer science majors who are juniors, and 1 computer science major who is a senior. Express each of the following statements in terms of quantifiers and then determine its truth value.

- **a)** There is a student in the class who is a junior.
- **b)** Every student in the class is a computer science major.
- **c)** There is a student in the class who is neither a mathematics major nor a junior.
- **d)** Every student in the class is either a sophomore or a computer science major.
- **e)** There is a major such that there is a student in the class in every year of study with that major.

19. Let $P(x)$ be the statement "$x = x^2$." If the universe of discourse is the set of integers, what are the truth values of the following?

a) $P(0)$ **b)** $P(1)$ **c)** $P(2)$
d) $P(-1)$ **e)** $\exists x P(x)$ **f)** $\forall x P(x)$

20. Let $Q(x, y)$ be the statement "$x + y = x - y$." If the universe of discourse for both variables is the set of integers, what are the truth values of the following?

a) $Q(1, 1)$ **b)** $Q(2, 0)$
c) $\forall y\, Q(1, y)$ **d)** $\exists x\, Q(x, 2)$
e) $\exists x \exists y\, Q(x, y)$ **f)** $\forall x \exists y\, Q(x, y)$
g) $\exists y \forall x\, Q(x, y)$ **h)** $\forall y \exists x\, Q(x, y)$
i) $\forall x \forall y\, Q(x, y)$

21. Determine the truth value of each of the following statements if the universe of discourse for all variables is the set of all integers.

a) $\forall n(n^2 \geq 0)$ **b)** $\exists n(n^2 = 2)$
c) $\forall n(n^2 \geq n)$ **d)** $\forall n \exists m(n^2 < m)$
e) $\exists n \forall m(n < m^2)$ **f)** $\forall n \exists m(n + m = 0)$
g) $\exists n \forall m(nm = m)$ **h)** $\exists n \exists m(n^2 + m^2 = 5)$
i) $\exists n \exists m(n^2 + m^2 = 6)$
j) $\exists n \exists m(n + m = 4 \land n - m = 1)$
k) $\exists n \exists m(n + m = 4 \land n - m = 2)$
l) $\forall n \forall m \exists p(p = (m + n)/2)$

22. Determine the truth value of each of the following statements if the universe of discourse of each variable is the set of real numbers.

a) $\exists x(x^2 = 2)$ **b)** $\exists x(x^2 = -1)$
c) $\forall x \exists y(x^2 = y)$ **d)** $\forall x \exists y(x = y^2)$
e) $\exists x \forall y(xy = 0)$ **f)** $\exists x \exists y(x + y \neq y + x)$
g) $\forall x \neq 0 \exists y(xy = 1)$ **h)** $\exists x \forall y \neq 0(xy = 1)$

i) $\forall x \exists y(x + y = 1)$
j) $\exists x \exists y(x + 2y = 2 \wedge 2x + 4y = 5)$
k) $\forall x \exists y(x + y = 2 \wedge 2x - y = 1)$
l) $\forall x \forall y \exists z(z = (x - y)/2)$

23. Suppose the universe of discourse of the propositional function $P(x, y)$ consists of pairs x and y, where x is 1, 2, or 3 and y is 1, 2, or 3. Write out the following propositions using disjunctions and conjunctions.

a) $\exists x P(x, 3)$
b) $\forall y P(1, y)$
c) $\forall x \forall y P(x, y)$
d) $\exists x \exists y P(x, y)$
e) $\exists x \forall y P(x, y)$
f) $\forall y \exists x P(x, y)$

24. Rewrite each of the following statements so that negations appear only within predicates (that is, so that no negation is outside a quantifier or an expression involving logical connectives).

a) $\neg \exists y \exists x P(x,y)$
b) $\neg \forall x \exists y P(x,y)$
c) $\neg \exists y (Q(y) \wedge \forall x \neg R(x,y))$
d) $\neg \exists y (\exists x R(x,y) \vee \forall x S(x,y))$
e) $\neg \exists y (\forall x \exists z T(x,y,z) \vee \exists x \forall z U(x,y,z))$

25. Rewrite each of the following statements so that negations appear only within predicates (that is, so that no negation is outside a quantifier or an expression involving logical connectives).

a) $\neg \forall x \forall y P(x,y)$
b) $\neg \forall y \exists x P(x,y)$
c) $\neg \forall y \forall x (P(x,y) \vee Q(x,y))$
d) $\neg(\exists x \exists y \neg P(x,y) \wedge \forall x \forall y Q(x,y))$
e) $\neg \forall x (\exists y \forall z P(x,y,z) \wedge \exists z \forall y P(x,y,z))$

26. Express each of the following statements using quantifiers. Then form the negation of the statement so that no negation is to the left of a quantifier. Next, express the negation in simple English. (Do not simply use the words "It is not the case that.")

a) All dogs have fleas.
b) No one has lost more than one thousand dollars playing the lottery.
c) There is a student in this class who has chatted with exactly one other student.
d) No student in this class has sent e-mail to exactly two other students in this class.
e) Some student has solved every exercise in this book.
f) No student has solved at least one exercise in every section of this book.

27. Express each of the following statements using quantifiers. Then form the negation of the statement, so that no negation is to the left of a quantifier. Next, express the negation in simple English. (Do not simply use the words "It is not the case that.")

a) There is no dog that can talk.
b) There is no one in this class who knows French and Russian.
c) Every student in this class has taken exactly two mathematics classes at this school.
d) Someone has visited every country in the world except Libya.
e) No one has climbed every mountain in the Himalayas.
f) Every movie actor has either been in a movie with Kevin Bacon or has been in a movie with someone who has been in a movie with Kevin Bacon.

28. Express the negations of the following propositions using quantifiers, and in English.

a) Every student in this class likes mathematics.
b) There is a student in this class who has never seen a computer.
c) There is a student in this class who has taken every mathematics course offered at this school.
d) There is a student in this class who has been in at least one room of every building on campus.

29. Use quantifiers to express the associative law for multiplication of real numbers.

30. Use quantifiers to express the distributive laws of multiplication over addition for real numbers.

Exercises 31–34 are based on questions found in the book *Symbolic Logic* by Lewis Carroll.

31. Let $P(x)$, $Q(x)$, and $R(x)$ be the statements "x is a professor," "x is ignorant," and "x is vain," respectively. Express each of the following statements using quantifiers; logical connectives; and $P(x)$, $Q(x)$, and $R(x)$, where the universe of discourse is the set of all people.

a) No professors are ignorant.
b) All ignorant people are vain.
c) No professors are vain.
d) Does (c) follow from (a) and (b)? If not, is there a correct conclusion?

32. Let $P(x)$, $Q(x)$, and $R(x)$ be the statements "x is a clear explanation," "x is satisfactory," and "x is an excuse," respectively. Suppose that the universe of discourse for x is the set of all English text. Express each of the following statements using quantifiers; logical connectives; and $P(x)$, $Q(x)$, and $R(x)$.

a) All clear explanations are satisfactory.
b) Some excuses are unsatisfactory.
c) Some excuses are not clear explanations.
***d)** Does (c) follow from (a) and (b)? If not, is there a correct conclusion?

33. Let $P(x)$, $Q(x)$, $R(x)$, and $S(x)$ be the statements "x is a baby," "x is logical," "x is able to manage a crocodile," and "x is despised," respectively. Suppose that the universe of discourse is the set of all people. Express each of the following statements using quantifiers; logical connectives; and $P(x)$, $Q(x)$, $R(x)$, and $S(x)$.

a) Babies are illogical.
b) Nobody is despised who can manage a crocodile.
c) Illogical persons are despised.
d) Babies cannot manage crocodiles.
***e)** Does (d) follow from (a), (b), and (c)? If not, is there a correct conclusion?

34. Let $P(x)$, $Q(x)$, $R(x)$, and $S(x)$ be the statements "x is a duck," "x is one of my poultry," "x is an officer," and "x is willing to waltz," respectively. Express each of the following statements using quantifiers; logical connectives; and $P(x)$, $Q(x)$, $R(x)$, and $S(x)$.

- **a)** No ducks are willing to waltz.
- **b)** No officers ever decline to waltz.
- **c)** All my poultry are ducks.
- **d)** My poultry are not officers.
- ***e)** Does (d) follow from (a), (b), and (c)? If not, is there a correct conclusion?

35. Show that the statements $\neg\exists x\forall y P(x,y)$ and $\forall x\exists y\neg P(x,y)$ have the same truth value.

36. Show that $\forall x(P(x)\wedge Q(x))$ and $\forall x\,P(x)\wedge\forall x\,Q(x)$ have the same truth value.

37. Show that $\exists x(P(x)\vee Q(x))$ and $\exists x\,P(x)\vee\exists x\,Q(x)$ have the same truth value.

38. Establish the following logical equivalences, where A is a proposition not involving any quantifiers.

- **a)** $(\forall x\,P(x))\vee A \iff \forall x(P(x)\vee A)$
- **b)** $(\exists x\,P(x))\vee A \iff \exists x(P(x)\vee A)$

39. Establish the following logical equivalences, where A is a proposition not involving any quantifiers.

- **a)** $(\forall x\,P(x))\wedge A \iff \forall x(P(x)\wedge A)$
- **b)** $(\exists x\,P(x))\wedge A \iff \exists x(P(x)\wedge A)$

40. Show that $\forall x\,P(x)\vee\forall x\,Q(x)$ and $\forall x\,(P(x)\vee Q(x))$ are not logically equivalent.

41. Show that $\exists x\,P(x)\wedge\exists x\,Q(x)$ and $\exists x(P(x)\wedge Q(x))$ are not logically equivalent.

***42.** Show that $\forall x\,P(x)\vee\forall x\,Q(x)$ and $\forall x\forall y\,(P(x)\vee Q(y))$ are logically equivalent. (The new variable y is used to combine the quantifications correctly.)

***43.** **a)** Show that $\forall x\,P(x)\wedge\exists x\,Q(x)$ and $\forall x\exists y\,(P(x)\wedge Q(y))$ are equivalent.

- **b)** Show that $\forall x\,P(x)\vee\exists x\,Q(x)$ and $\forall x\exists y\,(P(x)\vee Q(y))$ are equivalent.

44. The notation $\exists!x\,P(x)$ denotes the proposition

> "There exists a unique x such that $P(x)$ is true."

If the universe of discourse is the set of integers, what are the truth values of the following?

- **a)** $\exists!x(x>1)$
- **b)** $\exists!x(x^2=1)$
- **c)** $\exists!x(x+3=2x)$
- **d)** $\exists!x(x=x+1)$

45. What are the truth values of the following statements?

- **a)** $\exists!x\,P(x)\rightarrow\exists x\,P(x)$
- **b)** $\forall x\,P(x)\rightarrow\exists!x\,P(x)$
- **c)** $\exists!x\neg P(x)\rightarrow\neg\forall x\,P(x)$

46. Write out the quantification $\exists!x\,P(x)$, where the universe of discourse consists of the integers 1, 2, and 3, in terms of negations, conjunctions, and disjunctions.

***47.** Express the quantification $\exists!x\,P(x)$ using universal quantifications, existential quantifications, and logical operators.

A statement is in **prenex normal form (PNF)** if and only if it is of the form

$$Q_1x_1Q_2x_2\cdots Q_kx_kP(x_1,x_2,\ldots,x_k),$$

where each Q_i, $i=1,2,\ldots,k$, is either the existential quantifier or the universal quantifier, and $P(x_1,\ldots,x_k)$ is a predicate involving *no* quantifiers. For example, $\exists x\forall y(P(x,y)\wedge Q(y))$ is in prenex normal form, whereas $\exists x\,P(x)\vee\forall x\,Q(x)$ is not (since the quantifiers do not all occur first).

Every statement formed from propositional variables, predicates, **T**, and **F** using logical connectives and quantifiers is equivalent to a statement in prenex normal form. Exercise 49 asks for a proof of this fact.

***48.** Put the following statements in prenex normal form. (*Hint:* Use logical equivalence from Tables 5 and 6 in Section 1.2, Table 2 in this section, and Exercises 36–39 and 42–43 in this section.)

- **a)** $\exists x\,P(x)\vee\exists x\,Q(x)\vee A$, where A is a proposition not involving any quantifiers.
- **b)** $\neg(\forall x\,P(x)\vee\forall x\,Q(x))$
- **c)** $\exists x\,P(x)\rightarrow\exists x\,Q(x)$

****49.** Show how to transform an arbitrary statement to a statement in prenex normal form that is equivalent to the given statement.

A real number x is called an **upper bound** of a set S of real numbers if x is greater than or equal to every member of S. The real number x is called the **least upper bound** of a set S of real numbers if x is an upper bound of S and x is less than or equal to every upper bound of S; if the least upper bound of a set S exists, it is unique.

50. **a)** Using quantifiers, express the fact that x is an upper bound of S.

- **b)** Using quantifiers, express the fact that x is the least upper bound of S.

51. (Calculus required) Using quantifiers, express the fact that $\lim_{x\to a}f(x)$ does not exist.

The statement $\lim_{n\to\infty}a_n=L$ means that for every positive real number ϵ there is a positive integer N such that $|a_n-L|<\epsilon$ whenever $n>N$.

52. (Calculus required) Use quantifiers to express the statement that $\lim_{n\to\infty}a_n=L$.

53. (Calculus required) Use quantifiers to express the statement that $\lim_{n\to\infty}a_n$ does not exist.

54. (Calculus required) Use quantifiers to express the following definition: A sequence $\{a_n\}$ is a Cauchy sequence if for every real number $\epsilon>0$ there exists a positive integer N such that $|a_m-a_n|<\epsilon$ for every pair of positive integers m and n with $m>N$ and $n>N$.

55. (Calculus required) Use quantifiers and logical connectives to express the following definition: A number L is the **limit superior** of a sequence $\{a_n\}$ if for every real number $\epsilon>0$, $a_n>L-\epsilon$ for infinitely many n and $a_n>L+\epsilon$ for only finitely many n.

1.4

Sets

INTRODUCTION

We will study a wide variety of discrete structures in this book. These include relations, which consist of ordered pairs of elements; combinations, which are unordered collections of elements; and graphs, which are sets of vertices and edges connecting vertices. Moreover, we will illustrate how these and other discrete structures are used in modeling and problem solving. In particular, many examples of the use of discrete structures in the storage, communication, and manipulation of data will be described. In this section we study the fundamental discrete structure upon which all other discrete structures are built, namely, the set.

Sets are used to group objects together. Often, the objects in a set have similar properties. For instance, all the students who are currently enrolled in your school make up a set. Likewise, all the students currently taking a course in discrete mathematics at any school make up a set. In addition, those students enrolled in your school who are taking a course in discrete mathematics form a set that can be obtained by taking the elements common to the first two collections. The language of sets is a means to study such collections in an organized fashion.

web

Note that the term *object* has been used without specifying what an object is. This description of a set as a collection of objects, based on the intuitive notion of an object, was first stated by the German mathematician Georg Cantor in 1895. The theory that results from this intuitive definition of a set leads to **paradoxes,** or logical inconsistencies, as the English philosopher Bertrand Russell showed in 1902 (see Exercise 26 for a description of one of these paradoxes). These logical inconsistencies can be avoided by building set theory starting with basic assumptions, called **axioms.** We will use Cantor's original version of set theory, known as **naive set theory,** without developing an axiomatic version of set theory, since all sets considered in this book can be treated consistently using Cantor's original theory.

We now proceed with our discussion of sets.

DEFINITION 1. **The objects in a set are also called the *elements,* or *members,* of the set. A set is said to *contain* its elements.**

---

web

Georg Cantor (1845–1918). Georg Cantor was born in St. Petersburg, Russia, where his father was a successful merchant. Cantor developed his interest in mathematics in his teens. He began his university studies in Zurich in 1862, but when his father died he left Zurich. He continued his university studies at the University of Berlin in 1863, where he studied under the eminent mathematicians Weierstrass, Kummer, and Kronecker. He received his doctor's degree in 1867 after having written a dissertation on number theory. Cantor assumed a position at the University of Halle in 1869, where he continued working until his death.

Cantor is considered the founder of set theory. His contributions in this area include the discovery that the set of real numbers is uncountable. He is also noted for his many important contributions to analysis. Cantor also was interested in philosophy and wrote papers relating his theory of sets with metaphysics.

Cantor married in 1874 and had five children. His melancholy temperament was balanced by his wife's happy disposition. Although he received a large inheritance from his father, he was poorly paid as a professor. To mitigate this, he tried to obtain a better-paying position at the University of Berlin. His appointment there was blocked by Kronecker, who did not agree with Cantor's views on set theory. Cantor suffered from mental illness throughout the later years of his life. He died in 1918 in a psychiatric clinic.

There are several ways to describe a set. One way is to list all the members of a set, when this is possible. We use a notation where all members of the set are listed between braces. For example, the notation $\{a, b, c, d\}$ represents the set with the four elements a, b, c, and d.

EXAMPLE 1 The set V of all vowels in the English alphabet can written as $V = \{a, e, i, o, u\}$. ■

EXAMPLE 2 The set O of odd positive integers less than 10 can be expressed by $O = \{1, 3, 5, 7, 9\}$. ■

EXAMPLE 3 Although sets are usually used to group together elements with common properties, there is nothing that prevents a set from having seemingly unrelated elements. For instance, $\{a, 2,$ Fred, New Jersey$\}$ is the set containing the four elements a, 2, Fred, and New Jersey. ■

Uppercase letters are usually used to denote sets. The boldface letters $\mathbf{N}$, $\mathbf{Z}$, and $\mathbf{R}$ will be reserved to represent the set of natural numbers $\{0, 1, 2, 3, \ldots\}$, the set of integers $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$, and the set of real numbers, respectively. We will occasionally use the notation $\mathbf{Z}^+$ to denote the set of positive integers. (Some people do not consider 0 a natural number, so be careful to check how the term *natural numbers* is used when you read other books.)

Sometimes the brace notation is used to describe a set without listing all its members. Some members of the set are listed, and then *ellipses* ($\ldots$) are used when the general pattern of the elements is obvious.

EXAMPLE 4 The set of positive integers less than 100 can be denoted by $\{1, 2, 3, \ldots, 99\}$. ■

Since many mathematical statements assert that two differently specified collections of objects are really the same set, we need to understand what it means for two sets to be equal.

---

web **Bertrand Russell** (1872–1970). Bertrand Russell was born into a prominent English family active in the progressive movement and having a strong commitment to liberty. He became an orphan at an early age and was placed in the care of his father's parents, who had him educated at home. He entered Trinity College, Cambridge, in 1890, where he excelled in mathematics and in moral science. He won a fellowship on the basis of his work on the foundations of geometry. In 1910 Trinity College appointed him to a lectureship in logic and the philosophy of mathematics.

Russell fought for progressive causes throughout his life. He held strong pacifist views, and his protests against World War I led to dismissal from his position at Trinity College. He was imprisoned for 6 months in 1918 because of an article he wrote that was branded as seditious. Russell fought for women's suffrage in Great Britain. In 1961, at the age of 89, he was imprisoned for the second time for his protests advocating nuclear disarmament.

Russell's greatest work was in his development of principles that could be used as a foundation for all of mathematics. His most famous work is *Principia Mathematica*, written with Alfred North Whitehead, which attempts to deduce all of mathematics using a set of primitive axioms. He wrote many books on philosophy, physics, and his political ideas. Russell won the Nobel Prize for literature in 1950.

DEFINITION 2. **Two sets are *equal* if and only if they have the same elements.**

EXAMPLE 5 The sets $\{1, 3, 5\}$ and $\{3, 5, 1\}$ are equal, since they have the same elements. Note that the order in which the elements of a set are listed does not matter. Note also that it does not matter if an element of a set is listed more than once, so that $\{1, 3, 3, 3, 5, 5, 5, 5\}$ is the same as the set $\{1, 3, 5\}$ since they have the same elements. ■

Another way to describe a set is to use **set builder** notation. We characterize all those elements in the set by stating the property or properties they must have to be members. For instance, the set O of all odd positive integers less than 10 can be written as

$$O = \{x \mid x \text{ is an odd positive integer less than } 10\}.$$

We often use this type of notation to describe sets when it is impossible to list all the elements of the set. For instance, the set of all real numbers can be written as

$$\mathbf{R} = \{x \mid x \text{ is a real number}\}.$$

Sets can also be represented graphically using Venn diagrams, named after the English mathematician John Venn, who introduced their use in 1881. In Venn diagrams the **universal set** U, which contains all the objects under consideration, is represented by a rectangle. Inside this rectangle, circles or other geometrical figures are used to represent sets. Sometimes points are used to represent the particular elements of the set. Venn diagrams are often used to indicate the relationships between sets. We show how a Venn diagram can be used in the following example.

EXAMPLE 6 Draw a Venn diagram that represents V, the set of vowels in the English alphabet.

Solution: We draw a rectangle to indicate the universal set U, which is the set of the 26 letters of the English alphabet. Inside this rectangle we draw a circle to represent V. Inside this circle we indicate the elements of V with points (see Figure 1). ■

We will now introduce notation used to describe membership in sets. We write $a \in A$ to denote that a is an element of the set A. The notation $a \notin A$ denotes that a is not a member of the set A. Note that lowercase letters are usually used to denote elements of sets.

---

web

John Venn (1834–1923). John Venn was born into a London suburban family noted for its philanthropy. He attended London schools and got his mathematics degree from Caius College, Cambridge, in 1857. He was elected a fellow of this college and held his fellowship there until his death. He took holy orders in 1859 and, after a brief stint of religious work, returned to Cambridge, where he developed programs in the moral sciences. Besides his mathematical work, Venn had an interest in history and wrote extensively about his college and family.

Venn's book *Symbolic Logic* clarifies ideas originally presented by Boole. In this book, Venn presents a systematic development of a method that uses geometric figures, known now as *Venn diagrams*. Today these diagrams are primarily used to analyze logical arguments and to illustrate relationships between sets. In addition to his work on symbolic logic, Venn made contributions to probability theory described in his widely used textbook on that subject.

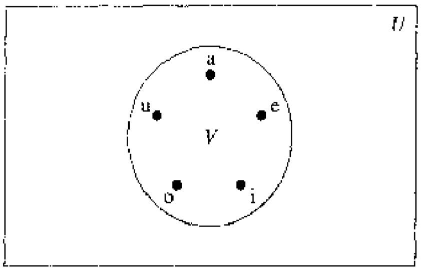


FIGURE 1 **Venn Diagram for the Set of Vowels.**

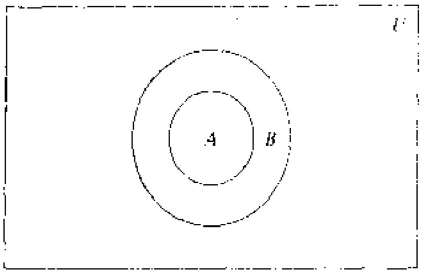


FIGURE 2 **Venn Diagram Showing That A Is a Subset of B.**

There is a special set that has no elements. This set is called the **empty set,** or **null set,** and is denoted by $\emptyset$. The empty set can also be denoted by $\{\ \}$ (that is, we represent the empty set with a pair of braces that encloses all the elements in this set). Often, a set of elements with certain properties turns out to be the null set. For instance, the set of all positive integers that are greater than their squares is the null set.

DEFINITION 3. **The set A is said to be a *subset* of B if and only if every element of A is also an element of B. We use the notation $A \subseteq B$ to indicate that A is a subset of the set B.**

We see that $A \subseteq B$ if and only if the quantification

$$\forall x(x \in A \rightarrow x \in B)$$

is true. For instance, the set of all odd positive integers less than 10 is a subset of the set of all positive integers less than 10. The set of all computer science majors at your school is a subset of the set of all students at your school.

The null set is a subset of every set, that is,

$$\emptyset \subseteq S$$

whenever S is a set. To establish that the null set is a subset of S, we must show that every element of the null set is also in S. In other words, we must show that the implication "if $x \in \emptyset$, then $x \in S$" is always true. We need only note that the hypothesis of this implication—namely, "$x \in \emptyset$"—is always false to see that this implication is always true. Hence, the empty set is a subset of every set. Furthermore, note that every set is a subset of itself (the reader should verify this). Consequently, if P is a set, we know that $\emptyset \subseteq P$ and $P \subseteq P$.

When we wish to emphasize that a set A is a subset of the set B but that $A \neq B$, we write $A \subset B$ and say that A is a **proper subset** of B. Venn diagrams can be used to show that a set A is a subset of a set B. We draw the universal set U as a rectangle. Within this rectangle we draw a circle for B. Since A is a subset of B, we draw the circle for A within the circle for B. This relationship is shown in Figure 2.

One way to show that two sets have the same elements is to show that each set is a subset of the other. In other words, we can show that if A and B are sets with $A \subseteq B$ and $B \subseteq A$, then $A = B$. This turns out to be a useful way to show that two sets are equal.

Sets may have other sets as members. For instance, we have the sets

$$\{\emptyset, \{a\}, \{b\}, \{a, b\}\} \qquad \text{and} \qquad \{x \mid x \text{ is a subset of the set } \{a, b\}\}.$$

Note that these two sets are equal.

Sets are used extensively in counting problems, and for such applications we need to discuss the size of sets.

DEFINITION 4. **Let S be a set. If there are exactly n distinct elements in S where n is a nonnegative integer, we say that S is a *finite set* and that n is the *cardinality* of S. The cardinality of S is denoted by $|S|$.**

EXAMPLE 7 Let A be the set of odd positive integers less than 10. Then $|A| = 5$. ■

EXAMPLE 8 Let S be the set of letters in the English alphabet. Then $|S| = 26$. ■

EXAMPLE 9 Since the null set has no elements, it follows that $|\emptyset| = 0$. ■

We will also be interested in sets that are not finite.

DEFINITION 5. **A set is said to be *infinite* if it is not finite.**

EXAMPLE 10 The set of positive integers is *infinite*. ■

The cardinality of infinite sets will be discussed in Section 1.7. In that section, we will discuss what it means for a set to be countable and show that certain sets are countable while others are not.

THE POWER SET

Many problems involve testing all combinations of elements of a set to see if they satisfy some property. To consider all such combinations of elements of a set S, we build a new set that has as its members all the subsets of S.

DEFINITION 6. **Given a set S, the *power set* of S is the set of all subsets of the set S. The power set of S is denoted by $P(S)$.**

EXAMPLE 11 What is the power set of the set $\{0, 1, 2\}$?

Solution: The power set $P(\{0, 1, 2\})$ is the set of all subsets of $\{0, 1, 2\}$. Hence,

$$P(\{0, 1, 2\}) = \{\emptyset, \{0\}, \{1\}, \{2\}, \{0, 1\}, \{0, 2\}, \{1, 2\}, \{0, 1, 2\}\}.$$

Note that the empty set and the set itself are members of this set of subsets. ■

EXAMPLE 12 What is the power set of the empty set? What is the power set of the set $\{\emptyset\}$?

Solution: The empty set has exactly one subset, namely, itself. Consequently,

$$P(\emptyset) = \{\emptyset\}.$$

The set $\{\emptyset\}$ has exactly two subsets, namely, $\emptyset$ and the set $\{\emptyset\}$ itself. Therefore,

$$P(\{\emptyset\}) = \{\emptyset, \{\emptyset\}\}.$$

■

If a set has n elements, then its power set has 2^n elements. We will demonstrate this fact in several ways in subsequent sections of the text.

CARTESIAN PRODUCTS

The order of elements in a collection is often important. Since sets are unordered, a different structure is needed to represent ordered collections. This is provided by **ordered n-tuples**.

DEFINITION 7. The *ordered n-tuple* $(a_1, a_2, \ldots, a_n)$ is the ordered collection that has a_1 as its first element, a_2 as its second element, ..., and a_n as its nth element.

We say that two ordered n-tuples are equal if and only if each corresponding pair of their elements is equal. In other words, $(a_1, a_2, \ldots, a_n) = (b_1, b_2, \ldots, b_n)$ if and only if $a_i = b_i$, for $i = 1, 2, \ldots, n$. In particular, 2-tuples are called **ordered pairs.** The ordered pairs (a, b) and (c, d) are equal if and only if $a = c$ and $b = d$. Note that (a, b) and (b, a) are not equal unless $a = b$.

Many of the discrete structures we will study in later chapters are based on the notion of the *Cartesian product* of sets (named after René Descartes). We first define the Cartesian product of two sets.

René Descartes (1596–1650). René Descartes was born into a noble family near Tours, France, about 200 miles southwest of Paris. He was the third child of his father's first wife; she died several days after his birth. Because of René's poor health, his father, a provincial judge, let his son's formal lessons slide until, at the age of 8, René entered the Jesuit college at La Flèche. The rector of the school took a liking to him and permitted him to stay in bed until late in the morning because of his frail health. From then on, Descartes spent his mornings in bed; he considered these times his most productive hours for thinking.

Descartes left school in 1612, moving to Paris, where he spent 2 years studying mathematics. He earned a law degree in 1616 from the University of Poitiers. At 18 Descartes became disgusted with studying and decided to see the world. He moved to Paris and became a successful gambler. However, he grew tired of bawdy living and moved to the suburb of Saint-Germain, where he devoted himself to mathematical study. When his gambling friends found him, he decided to leave France and undertake a military career. However, he never did any fighting. One day, while escaping the cold in an overheated room at a military encampment, he had several feverish dreams, which revealed his future career as a mathematician and philosopher.

After ending his military career, he traveled throughout Europe. He then spent several years in Paris, where he studied mathematics and philosophy and constructed optical instruments. Descartes decided to move to Holland, where he spent 20 years wandering around the country, accomplishing his most important work. During this time he wrote several books, including the *Discours*, which contains his contributions to analytic geometry, for which he is best known. He also made fundamental contributions to philosophy.

In 1649 Descartes was invited by Queen Christina to visit her court in Sweden to tutor her in philosophy. Although he was reluctant to live in what he called "the land of bears amongst rocks and ice," he finally accepted the invitation and moved to Sweden. Unfortunately, the winter of 1649–1650 was extremely bitter. Descartes caught pneumonia and died in mid-February.

DEFINITION 8. **Let A and B be sets. The *Cartesian product* of A and B, denoted by $A \times B$, is the set of all ordered pairs (a, b) where $a \in A$ and $b \in B$. Hence,**

$$A \times B = \{(a, b) \mid a \in A \wedge b \in B\}.$$

EXAMPLE 13 Let A represent the set of all students at a university, and let B represent the set of all courses offered at the university. What is the Cartesian product $A \times B$?

Solution: The Cartesian product $A \times B$ consists of all the ordered pairs of the form (a, b), where a is a student at the university and b is a course offered at the university. The set $A \times B$ can be used to represent all possible enrollments of students in courses at the university. ■

EXAMPLE 14 What is the Cartesian product of $A = \{1, 2\}$ and $B = \{a, b, c\}$?

Solution: The Cartesian product $A \times B$ is

$$A \times B = \{(1, a), (1, b), (1, c), (2, a), (2, b), (2, c)\}.$$ ■

The Cartesian products $A \times B$ and $B \times A$ are not equal, unless $A = \emptyset$ or $B = \emptyset$ (so that $A \times B = \emptyset$) or unless $A = B$ (see Exercise 24, at the end of this section). This is illustrated in the following example.

EXAMPLE 15 Show that the Cartesian product $B \times A$ is not equal to the Cartesian product $A \times B$, where A and B are as in Example 14.

Solution: The Cartesian product $B \times A$ is

$$B \times A = \{(a, 1), (a, 2), (b, 1), (b, 2), (c, 1), (c, 2)\}.$$

This is not equal to $A \times B$, which was found in Example 14. ■

The Cartesian product of more than two sets can also be defined.

DEFINITION 9. **The *Cartesian product* of the sets $A_1, A_2, \ldots, A_n$, denoted by $A_1 \times A_2 \times \cdots \times A_n$, is the set of ordered n-tuples $(a_1, a_2, \ldots, a_n)$, where a_i belongs to A_i for $i = 1, 2, \ldots, n$. In other words**

$$A_1 \times A_2 \times \cdots \times A_n = \{(a_1, a_2, \ldots, a_n) \mid a_i \in A_i \quad \text{for } i = 1, 2, \ldots, n\}.$$

EXAMPLE 16 What is the Cartesian product $A \times B \times C$, where $A = \{0, 1\}$, $B = \{1, 2\}$, and $C = \{0, 1, 2\}$?

Solution: The Cartesian product $A \times B \times C$ consists of all ordered triples (a, b, c), where $a \in A$, $b \in B$, and $c \in C$. Hence,

$$A \times B \times C = \{(0,1,0), (0,1,1), (0,1,2), (0,2,0), (0,2,1), (0,2,2), (1,1,0), (1,1,1), (1,1,2), (1,2,0), (1,2,1), (1,2,2)\}.$$ ■

Exercises

1. List the members of the following sets.
   - **a)** $\{x \mid x \text{ is a real number such that } x^2 = 1\}$
   - **b)** $\{x \mid x \text{ is a positive integer less than } 12\}$
   - **c)** $\{x \mid x \text{ is the square of an integer and } x < 100\}$
   - **d)** $\{x \mid x \text{ is an integer such that } x^2 = 2\}$
2. Use set builder notation to give a description of each of the following sets.
   - **a)** $\{0, 3, 6, 9, 12\}$
   - **b)** $\{-3, -2, -1, 0, 1, 2, 3\}$
   - **c)** $\{m, n, o, p\}$
3. Determine whether each of the following pairs of sets is equal.
   - **a)** $\{1, 3, 3, 3, 5, 5, 5, 5, 5\}, \{5, 3, 1\}$
   - **b)** $\{\{1\}\}, \{1, \{1\}\}$
   - **c)** $\emptyset, \{\emptyset\}$
4. Suppose that $A = \{2, 4, 6\}$, $B = \{2, 6\}$, $C = \{4, 6\}$, and $D = \{4, 6, 8\}$. Determine which of these sets are subsets of which other of these sets.
5. For each of the following sets, determine whether 2 is an element of that set.
   - **a)** $\{x \in \mathbf{R} \mid x \text{ is an integer greater than } 1\}$
   - **b)** $\{x \in \mathbf{R} \mid x \text{ is the square of an integer}\}$
   - **c)** $\{2, \{2\}\}$
   - **d)** $\{\{2\}, \{\{2\}\}\}$
   - **e)** $\{\{2\}, \{2, \{2\}\}\}$
   - **f)** $\{\{\{2\}\}\}$
6. For each of the sets in Exercise 5, determine whether $\{2\}$ is an element of that set.
7. Determine whether each of the following statements is true or false.
   - **a)** $x \in \{x\}$
   - **b)** $\{x\} \subseteq \{x\}$
   - **c)** $\{x\} \in \{x\}$
   - **d)** $\{x\} \in \{\{x\}\}$
   - **e)** $\emptyset \subseteq \{x\}$
   - **f)** $\emptyset \in \{x\}$
8. Use a Venn diagram to illustrate the relationship $A \subseteq B$ and $B \subset C$.
9. Suppose that A, B, and C are sets such that $A \subseteq B$ and $B \subseteq C$. Show that $A \subseteq C$.
10. Find two sets A and B such that $A \in B$ and $A \subseteq B$.
11. What is the cardinality of each of the following sets?
    - **a)** $\{a\}$
    - **b)** $\{\{a\}\}$
    - **c)** $\{a, \{a\}\}$
    - **d)** $\{a, \{a\}, \{a, \{a\}\}\}$
12. What is the cardinality of each of the following sets?
    - **a)** $\emptyset$
    - **b)** $\{\emptyset\}$
    - **c)** $\{\emptyset, \{\emptyset\}\}$
    - **d)** $\{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$
13. Find the power set of each of the following sets.
    - **a)** $\{a\}$
    - **b)** $\{a, b\}$
    - **c)** $\{\emptyset, \{\emptyset\}\}$
14. Can you conclude that $A = B$ if A and B are two sets with the same power set?
15. How many elements does each of the following sets have?
    - **a)** $P(\{a, b, \{a, b\}\})$
    - **b)** $P(\{\emptyset, a, \{a\}, \{\{a\}\}\})$
    - **c)** $P(P(\emptyset))$
16. Determine whether each of the following sets is the power set of a set.
    - **a)** $\emptyset$
    - **b)** $\{\emptyset, \{a\}\}$
    - **c)** $\{\emptyset, \{a\}, \{\emptyset, a\}\}$
    - **d)** $\{\emptyset, \{a\}, \{b\}, \{a, b\}\}$
17. Let $A = \{a, b, c, d\}$ and $B = \{y, z\}$. Find
    - **a)** $A \times B$
    - **b)** $B \times A$
18. What is the Cartesian product $A \times B$, where A is the set of courses offered by the mathematics department at a university and B is the set of mathematics professors at this university?
19. What is the Cartesian product $A \times B \times C$, where A is the set of all airlines and B and C are both the set of all cities in the United States?
20. Suppose that $A \times B = \emptyset$, where A and B are sets. What can you conclude?
21. Let A be a set. Show that $\emptyset \times A = A \times \emptyset = \emptyset$.
22. Let $A = \{a, b, c\}$, $B = \{x, y\}$, and $C = \{0, 1\}$. Find
    - **a)** $A \times B \times C$
    - **b)** $C \times B \times A$
    - **c)** $C \times A \times B$
    - **d)** $B \times B \times B$
23. How many different elements does $A \times B$ have if A has m elements and B has n elements?
24. Show that $A \times B \neq B \times A$, when A and B are nonempty unless $A = B$.

*25. Show that the ordered pair (a, b) can be defined in terms of sets as $\{\{a\}, \{a, b\}\}$. (*Hint:* First show that $\{\{a\}, \{a, b\}\} = \{\{c\}, \{c, d\}\}$ if and only if $a = c$ and $b = d$.)

*26. *web* In this exercise **Russell's paradox** is presented. Let S be the set that contains a set x if the set x does not belong to itself, so that $S = \{x \mid x \notin x\}$.
   - **a)** Show that the assumption that S is a member of S leads to a contradiction.
   - **b)** Show that the assumption that S is not a member of S leads to a contradiction.

   From parts (a) and (b) it follows that the set S cannot be defined as it was. This paradox can be avoided by restricting the types of elements that sets can have.

*27. Describe a procedure for listing all the subsets of a finite set.

1.5

Set Operations

INTRODUCTION

Two sets can be combined in many different ways. For instance, starting with the set of mathematics majors and the set of computer science majors at your school, we can form the set of students who are mathematics majors or computer science majors, the set of students who are joint majors in mathematics and computer science, the set of all students not majoring in mathematics, and so on.

DEFINITION 1. Let A and B be sets. The *union* of the sets A and B, denoted by $A \cup B$, is the set that contains those elements that are either in A or in B, or in both.

An element x belongs to the union of the sets A and B if and only if x belongs to A or x belongs to B. This tells us that

$$A \cup B = \{x \mid x \in A \vee x \in B\}.$$

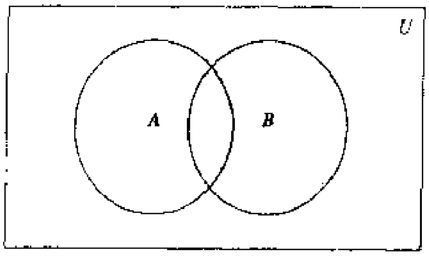
The Venn diagram shown in Figure 1 represents the union of two sets A and B. The shaded area within the circle representing A or the circle representing B is the area that represents the union of A and B.

We will give some examples of the union of sets.

EXAMPLE 1 The union of the sets $\{1, 3, 5\}$ and $\{1, 2, 3\}$ is the set $\{1, 2, 3, 5\}$; that is, $\{1, 3, 5\} \cup \{1, 2, 3\} = \{1, 2, 3, 5\}$. ■

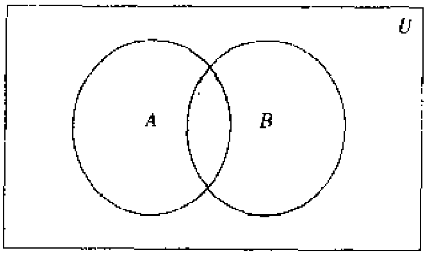
EXAMPLE 2 The union of the set of all computer science majors at your school and the set of all mathematics majors at your school is the set of students at your school who are majoring either in mathematics or in computer science (or in both). ■

DEFINITION 2. Let A and B be sets. The *intersection* of the sets A and B, denoted by $A \cap B$, is the set containing those elements in both A and B.



$A \cup B$ is shaded.

FIGURE 1 Venn Diagram Representing the Union of A and B.



$A \cap B$ is shaded.

FIGURE 2 Venn Diagram Representing the Intersection of A and B.

An element x belongs to the intersection of the sets A and B if and only if x belongs to A and x belongs to B. This tells us that

$$A \cap B = \{x \mid x \in A \wedge x \in B\}.$$

The Venn diagram shown in Figure 2 represents the intersection of two sets A and B. The shaded area that is within both the circles representing the sets A and B is the area that represents the intersection of A and B.

We give some examples of the intersection of sets.

EXAMPLE 3 The intersection of the sets $\{1, 3, 5\}$ and $\{1, 2, 3\}$ is the set $\{1, 3\}$; that is, $\{1, 3, 5\} \cap \{1, 2, 3\} = \{1, 3\}$. ■

EXAMPLE 4 The intersection of the set of all computer science majors at your school and the set of all mathematics majors is the set of all students who are joint majors in mathematics and computer science. ■

DEFINITION 3. Two sets are called *disjoint* if their intersection is the empty set.

EXAMPLE 5 Let $A = \{1, 3, 5, 7, 9\}$ and $B = \{2, 4, 6, 8, 10\}$. Since $A \cap B = \emptyset$, A and B are disjoint. ■

We often are interested in finding the cardinality of the union of sets. To find the number of elements in the union of two finite sets A and B, note that $|A| + |B|$ counts each element that is in A but not in B or in B but not in A exactly once, and each element that is in both A and B exactly twice. Thus, if the number of elements that are in both A and B is subtracted from $|A| + |B|$, elements in $A \cap B$ will be counted only once. Hence,

$$|A \cup B| = |A| + |B| - |A \cap B|.$$

The generalization of this result to unions of an arbitrary number of sets is called the **principle of inclusion–exclusion.** The principle of inclusion–exclusion is an important technique used in the art of enumeration. We will discuss this principle and other counting techniques in detail in Chapters 4 and 5.

There are other important ways to combine sets.

DEFINITION 4. Let A and B be sets. The *difference* of A and B, denoted by $A - B$, is the set containing those elements that are in A but not in B. The difference of A and B is also called the *complement of B with respect to A*.

An element x belongs to the difference of A and B if and only if $x \in A$ and $x \notin B$. This tells us that

$$A - B = \{x \mid x \in A \wedge x \notin B\}.$$

The Venn diagram shown in Figure 3 represents the difference of the sets A and B. The shaded area inside the circle that represents A and outside the circle that represents B is the area that represents $A - B$.

We give some examples of differences of sets.

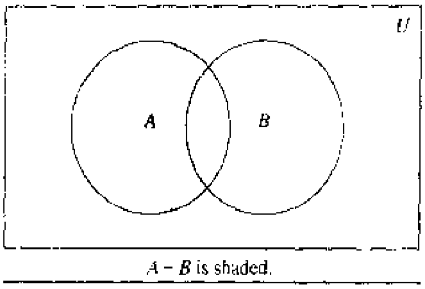


FIGURE 3 Venn Diagram for the Difference of A and B.

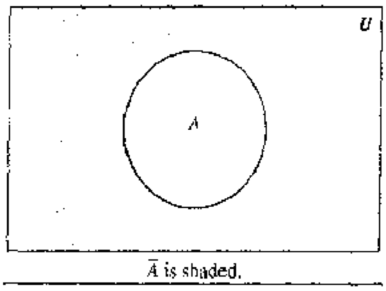


FIGURE 4 Venn Diagram for the Complement of the Set A.

EXAMPLE 6 The difference of $\{1, 3, 5\}$ and $\{1, 2, 3\}$ is the set $\{5\}$; that is, $\{1, 3, 5\} - \{1, 2, 3\} = \{5\}$. This is different from the difference of $\{1, 2, 3\}$ and $\{1, 3, 5\}$, which is the set $\{2\}$. ■

EXAMPLE 7 The difference of the set of computer science majors at your school and the set of mathematics majors at your school is the set of all computer science majors at your school who are not also mathematics majors. ■

Once the universal set U has been specified, the **complement** of a set can be defined.

DEFINITION 5. **Let U be the universal set. The *complement* of the set A, denoted by $\overline{A}$, is the complement of A with respect to U. In other words, the complement of the set A is $U - A$.**

An element belongs to $\overline{A}$ if and only if $x \notin A$. This tells us that

$$\overline{A} = \{x \mid x \notin A\}.$$

In Figure 4 the shaded area outside the circle that represents A is the area representing $\overline{A}$.

We give some examples of the complement of a set.

EXAMPLE 8 Let $A = \{a, e, i, o, u\}$ (where the universal set is the set of letters of the English alphabet). Then $\overline{A} = \{b, c, d, f, g, h, j, k, l, m, n, p, q, r, s, t, v, w, x, y, z\}$. ■

EXAMPLE 9 Let A be the set of positive integers greater than 10 (with universal set the set of all positive integers). Then $\overline{A} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. ■

SET IDENTITIES

Table 1 lists the most important set identities. We will prove several of these identities here, using three different methods. These methods are presented to illustrate that there are often many different approaches to the solution of a problem. The proofs of

TABLE 1 Set Identities.

Identity	*Name*
$A \cup \emptyset = A$ $A \cap U = A$	Identity laws
$A \cup U = U$ $A \cap \emptyset = \emptyset$	Domination laws
$A \cup A = A$ $A \cap A = A$	Idempotent laws
$\overline{(\overline{A})} = A$	Complementation law
$A \cup B = B \cup A$ $A \cap B = B \cap A$	Commutative laws
$A \cup (B \cup C) = (A \cup B) \cup C$ $A \cap (B \cap C) = (A \cap B) \cap C$	Associative laws
$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	Distributive laws
$\overline{A \cup B} = \overline{A} \cap \overline{B}$ $\overline{A \cap B} = \overline{A} \cup \overline{B}$	De Morgan's laws

the remaining identities will be left as exercises. The reader should note the similarity between these set identities and the logical equivalences discussed in Section 1.2. In fact, the set identities given can be proved directly from the corresponding logical equivalences. Furthermore, both are special cases of identities that hold for Boolean algebra (discussed in Chapter 9).

One way to prove that two sets are equal is to show that one of the sets is a subset of the other and vice versa. We illustrate this type of proof by establishing the second of De Morgan's laws.

EXAMPLE 10 Prove that $\overline{A \cap B} = \overline{A} \cup \overline{B}$ by showing that each set is a subset of the other.

Solution: First, suppose that $x \in \overline{A \cap B}$. It follows that $x \notin A \cap B$. This implies that $x \notin A$ or $x \notin B$. Hence, $x \in \overline{A}$ or $x \in \overline{B}$. Thus, $x \in \overline{A} \cup \overline{B}$. This shows that $\overline{A \cap B} \subseteq \overline{A} \cup \overline{B}$.

Now suppose that $x \in \overline{A} \cup \overline{B}$. Then $x \in \overline{A}$ or $x \in \overline{B}$. It follows that $x \notin A$ or $x \notin B$. Hence, $x \notin A \cap B$. Therefore, $x \in \overline{A \cap B}$. This demonstrates that $\overline{A} \cup \overline{B} \subseteq \overline{A \cap B}$. Since we have demonstrated that each set is a subset of the other, these two sets must be equal and the identity is proved. ■

Another way to verify set identities is to use set builder notation and the rules of logic. Consider the following proof of the second of De Morgan's laws.

EXAMPLE 11 Use set builder notation and logical equivalences to show that $\overline{A \cap B} = \overline{A} \cup \overline{B}$.

Solution: The following chain of equalities provides a demonstration of this identity:

$$\begin{aligned}
\overline{A \cap B} &= \{x \mid x \notin A \cap B\} \\
&= \{x \mid \neg(x \in (A \cap B))\} \\
&= \{x \mid \neg(x \in A \wedge x \in B)\} \\
&= \{x \mid x \notin A \vee x \notin B\} \\
&= \{x \mid x \in \overline{A} \vee x \in \overline{B}\} \\
&= \{x \mid x \in \overline{A} \cup \overline{B}\}.
\end{aligned}$$

Note that the second De Morgan's law for logical equivalences was used in the fourth equality of this chain. ■

Set identities can also be proved using **membership tables.** We consider each combination of sets that an element can belong to and verify that elements in the same combinations of sets belong to both the sets in the identity. To indicate that an element is in a set, a 1 is used; to indicate that an element is not in a set, a 0 is used. (The reader should note the similarity between membership tables and truth tables.)

EXAMPLE 12 Use a membership table to show that $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

Solution: The membership table for these combinations of sets is shown in Table 2. This table has eight rows. Since the columns for $A \cap (B \cup C)$ and $(A \cap B) \cup (A \cap C)$ are the same, the identity is valid. ■

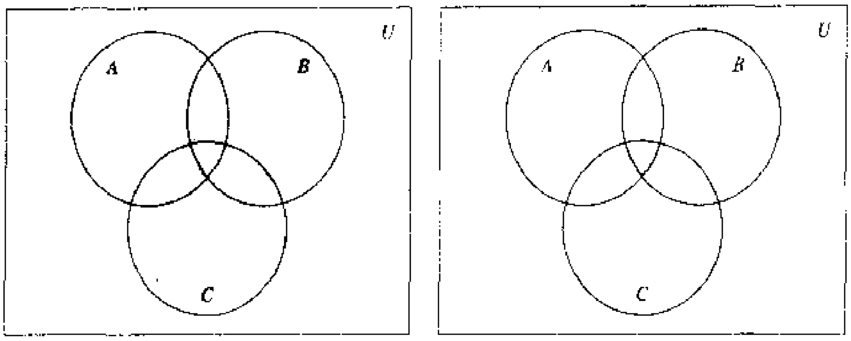
Additional set identities can be established using those that we have already proved. Consider the following example.

EXAMPLE 13 Let A, B, and C be sets. Show that

$$\overline{A \cup (B \cap C)} = (\overline{C} \cup \overline{B}) \cap \overline{A}.$$

TABLE 2 A Membership Table for the Distributive Property.

A	B	C	$B \cup C$	$A \cap (B \cup C)$	$A \cap B$	$A \cap C$	$(A \cap B) \cup (A \cap C)$
1	1	1	1	1	1	1	1
1	1	0	1	1	1	0	1
1	0	1	1	1	0	1	1
1	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0



(a) $A \cup B \cup C$ is shaded.

(b) $A \cap B \cap C$ is shaded.

FIGURE 5 (a) $A \cup B \cup C$ Is Shaded. (b) $A \cap B \cap C$ Is Shaded. The Union and Intersection of A, B, and C.

Solution: We have

$$\begin{aligned}\overline{A \cup (B \cap C)} &= \overline{A} \cap \overline{(B \cap C)} && \text{by the first De Morgan's law}\\ &= \overline{A} \cap (\overline{B} \cup \overline{C}) && \text{by the second De Morgan's law}\\ &= (\overline{B} \cup \overline{C}) \cap \overline{A} && \text{by the commutative law for intersections}\\ &= (\overline{C} \cup \overline{B}) \cap \overline{A} && \text{by the commutative law for unions.}\end{aligned}$$

■

GENERALIZED UNIONS AND INTERSECTIONS

Since unions and intersections of sets satisfy associative laws, the sets $A \cup B \cup C$ and $A \cap B \cap C$ are well defined when A, B, and C are sets. Note that $A \cup B \cup C$ contains those elements that are in at least one of the sets A, B, and C, and that $A \cap B \cap C$ contains those elements that are in all of A, B, and C. These combinations of the three sets, A, B, and C, are shown in Figure 5.

EXAMPLE 14 Let $A = \{0, 2, 4, 6, 8\}$, $B = \{0, 1, 2, 3, 4\}$, and $C = \{0, 3, 6, 9\}$. What are $A \cup B \cup C$ and $A \cap B \cap C$?

Solution: The set $A \cup B \cup C$ contains those elements in at least one of A, B, and C. Hence,

$$A \cup B \cup C = \{0, 1, 2, 3, 4, 6, 8, 9\}.$$

The set $A \cap B \cap C$ contains those elements in all three of A, B, and C. Thus,

$$A \cap B \cap C = \{0\}.$$

■

We can also consider unions and intersections of an arbitrary number of sets. We use the following definitions.

DEFINITION 6. The ***union*** **of a collection of sets is the set that contains those elements that are members of at least one set in the collection.**

We use the notation

$$A_1 \cup A_2 \cup \cdots \cup A_n = \bigcup_{i=1}^{n} A_i$$

to denote the union of the sets $A_1, A_2, \ldots, A_n$.

DEFINITION 7. The *intersection* of a collection of sets is the set that contains those elements that are members of all the sets in the collection.

We use the notation

$$A_1 \cap A_2 \cap \cdots \cap A_n = \bigcap_{i=1}^{n} A_i$$

to denote the intersection of the sets $A_1, A_2, \ldots, A_n$. We illustrate generalized unions and intersections with the following example.

EXAMPLE 15 Let $A_i = \{i, i+1, i+2, \ldots\}$. Then

$$\bigcup_{i=1}^{n} A_i = \bigcup_{i=1}^{n} \{i, i+1, i+2, \ldots\} = \{1, 2, 3, \ldots\},$$

and

$$\bigcap_{i=1}^{n} A_i = \bigcap_{i=1}^{n} \{i, i+1, i+2, \ldots\} = \{n, n+1, n+2, \ldots\}.$$ ■

COMPUTER REPRESENTATION OF SETS

There are various ways to represent sets using a computer. One method is to store the elements of the set in an unordered fashion. However, if this is done, the operations of computing the union, intersection, or difference of two sets would be time-consuming, since each of these operations would require a large amount of searching for elements. We will present a method for storing elements using an arbitrary ordering of the elements of the universal set. This method of representing sets makes computing combinations of sets easy.

Assume that the universal set U is finite (and of reasonable size so that the number of elements of U is not larger than the memory size of the computer being used). First, specify an arbitrary ordering of the elements of U, for instance $a_1, a_2, \ldots, a_n$. Represent a subset A of U with the bit string of length n, where the ith bit in this string is 1 if a_i belongs to A and is 0 if a_i does not belong to A. The following example illustrates this technique.

EXAMPLE 16 Let $U = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$, and the ordering of elements of U has the elements in increasing order; i.e., $a_i = i$. What bit strings represent the subset of all odd integers in U, the subset of all even integers in U, and the subset of integers not exceeding 5 in U?

Solution: The bit string that represents the set of odd integers in U, namely, $\{1, 3, 5, 7, 9\}$, has a one bit in the first, third, fifth, seventh, and ninth positions, and a zero elsewhere. It is

10 1010 1010.

(We have split this bit string of length 10 into two blocks of length five for easy reading since long bit strings are difficult to read.) Similarly, we represent the subset of all even integers in U, namely, $\{2, 4, 6, 8, 10\}$, by the string

01 0101 0101.

The set of all integers in U that do not exceed 5, namely, $\{1, 2, 3, 4, 5\}$, is represented by the string

11 1110 0000. ■

Using bit strings to represent sets, it is easy to find complements of sets and unions, intersections, and differences of sets. To find the bit string for the complement of a set from the bit string for that set, we simply change each 1 to a 0 and each 0 to 1, since $x \in A$ if and only if $x \notin \bar{A}$. Note that this operation corresponds to taking the negation of each bit when we associate a bit with a truth value—with 1 representing true and 0, false.

EXAMPLE 17 We have seen that the bit string for the set $\{1, 3, 5, 7, 9\}$ (with universal set $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$) is

10 1010 1010.

What is the bit string for the complement of this set?

Solution: The bit string for the complement of this set is obtained by replacing 0s with 1s and vice versa. This yields the string

01 0101 0101,

which corresponds to the set $\{2, 4, 6, 8, 10\}$. ■

To obtain the bit string for the union and intersection of two sets we perform bitwise Boolean operations on the bit strings representing the two sets. The bit in the ith position of the bit string of the union is 1 if either of the bits in the ith position in the two strings is 1 (or both are 1) and is 0 when both bits are 0. Hence, the bit string for the union is the bitwise *OR* of the bit strings for the two sets. The bit in the ith position of the bit string of the intersection is 1 when the bits in the corresponding position in the two strings are both 1 and is 0 when either of the two bits is 0 (or both are). Hence, the bit string for the intersection is the bitwise *AND* of the bit strings for the two sets.

EXAMPLE 18 The bit strings for the sets $\{1, 2, 3, 4, 5\}$ and $\{1, 3, 5, 7, 9\}$ are 11 1110 0000 and 10 1010 1010, respectively. Use bit strings to find the union and intersection of these sets.

Solution: The bit string for the union of these sets is

$$11\ 1110\ 0000 \vee 10\ 1010\ 1010 = 11\ 1110\ 1010,$$

which corresponds to the set $\{1, 2, 3, 4, 5, 7, 9\}$. The bit string for the intersection of these sets is

$$11\ 1110\ 0000 \wedge 10\ 1010\ 1010 = 10\ 1010\ 0000,$$

which corresponds to the set $\{1, 3, 5\}$. ■

Exercises

1. Let A be the set of students who live within one mile of school and let B be the set of students who walk to classes. Describe the students in each of the following sets.
  a) $A \cap B$ **b)** $A \cup B$
  c) $A - B$ **d)** $B - A$

2. Suppose that A is the set of sophomores at your school and B is the set of students in discrete mathematics at your school. Express each of the following sets in terms of A and B.
  a) the set of sophomores taking discrete mathematics in your school
  b) the set of sophomores at your school who are not taking discrete mathematics
  c) the set of students at your school who either are sophomores or are taking discrete mathematics
  d) the set of students at your school who either are not sophomores or are not taking discrete mathematics

3. Let $A = \{1, 2, 3, 4, 5\}$ and $B = \{0, 3, 6\}$. Find
  a) $A \cup B$. **b)** $A \cap B$.
  c) $A - B$. **d)** $B - A$.

4. Let $A = \{a, b, c, d, e\}$ and $B = \{a, b, c, d, e, f, g, h\}$. Find
  a) $A \cup B$. **b)** $A \cap B$.
  c) $A - B$. **d)** $B - A$.

5. Let A be a set. Show that $\overline{\overline{A}} = A$.

6. Let A be a set. Show that
  a) $A \cup \emptyset = A$. **b)** $A \cap \emptyset = \emptyset$.
  c) $A \cup A = A$. **d)** $A \cap A = A$.
  e) $A - \emptyset = A$. **f)** $A \cup U = U$.
  g) $A \cap U = A$. **h)** $\emptyset - A = \emptyset$.

7. Let A and B be sets. Show that
  a) $A \cup B = B \cup A$. **b)** $A \cap B = B \cap A$.

8. Find the sets A and B if $A - B = \{1, 5, 7, 8\}$, $B - A = \{2, 10\}$, and $A \cap B = \{3, 6, 9\}$.

9. Show that if A and B are sets, then $\overline{A \cup B} = \overline{A} \cap \overline{B}$.
  a) by showing each side is a subset of the other side
  b) using a membership table

10. Let A and B be sets. Show that
  a) $(A \cap B) \subseteq A$. **b)** $A \subseteq (A \cup B)$.
  c) $A - B \subseteq A$. **d)** $A \cap (B - A) = \emptyset$.
  e) $A \cup (B - A) = A \cup B$.

11. Show that if A, B, and C are sets, then $\overline{A \cap B \cap C} = \overline{A} \cup \overline{B} \cup \overline{C}$
  a) by showing each side is a subset of the other side.
  b) using a membership table.

12. Let A, B, and C be sets. Show that
  a) $(A \cup B) \subseteq (A \cup B \cup C)$.
  b) $(A \cap B \cap C) \subseteq (A \cap B)$.
  c) $(A - B) - C \subseteq A - C$.
  d) $(A - C) \cap (C - B) = \emptyset$.
  e) $(B - A) \cup (C - A) = (B \cup C) - A$.

13. Show that if A and B are sets, then $A - B = A \cap \overline{B}$.

14. Show that if A and B are sets, then $(A \cap B) \cup (A \cap \overline{B}) = A$.

15. Let A, B, and C be sets. Show that
  a) $A \cup (B \cup C) = (A \cup B) \cup C$.
  b) $A \cap (B \cap C) = (A \cap B) \cap C$.
  c) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$.

16. Let A, B, and C be sets. Show that $(A - B) - C = (A - C) - (B - C)$.

17. Let $A = \{0, 2, 4, 6, 8, 10\}$, $B = \{0, 1, 2, 3, 4, 5, 6\}$, and $C = \{4, 5, 6, 7, 8, 9, 10\}$. Find
  a) $A \cap B \cap C$. **b)** $A \cup B \cup C$.
  c) $(A \cup B) \cap C$. **d)** $(A \cap B) \cup C$.

18. Draw the Venn diagrams for each of the following combinations of the sets A, B, and C.
  a) $A \cap (B \cup C)$ **b)** $\overline{A} \cap \overline{B} \cap \overline{C}$
  c) $(A - B) \cup (A - C) \cup (B - C)$

19. What can you say about the sets A and B if the following are true?
  a) $A \cup B = A$ **b)** $A \cap B = A$
  c) $A - B = A$ **d)** $A \cap B = B \cap A$
  e) $A - B = B - A$

20. Can you conclude that $A = B$ if A, B, and C are sets such that
  a) $A \cup C = B \cup C$?
  b) $A \cap C = B \cap C$?

21. Let A and B be subsets of a universal set U. Show that $A \subseteq B$ if and only if $\overline{B} \subseteq \overline{A}$.

The **symmetric difference** of A and B, denoted by $A \oplus B$, is the set containing those elements in either A or B, but not in both A and B.

22. Find the symmetric difference of $\{1, 3, 5\}$ and $\{1, 2, 3\}$.

23. Find the symmetric difference of the set of computer science majors at a school and the set of mathematics majors at this school.

24. Draw a Venn diagram for the symmetric difference of the sets A and B.

25. Show that $A \oplus B = (A \cup B) - (A \cap B)$.

26. Show that $A \oplus B = (A - B) \cup (B - A)$.

27. Show that if A is a subset of a universal set U, then

a) $A \oplus A = \emptyset$. **b)** $A \oplus \emptyset = A$.
c) $A \oplus U = \overline{A}$. **d)** $A \oplus \overline{A} = U$.

28. Show that if A and B are sets, then

a) $A \oplus B = B \oplus A$. **b)** $(A \oplus B) \oplus B = A$.

29. What can you say about the sets A and B if $A \oplus B = A$?

***30.** Determine whether the symmetric difference is associative; that is, if A, B, and C are sets, does it follow that

$$A \oplus (B \oplus C) = (A \oplus B) \oplus C?$$

***31.** Suppose that A, B, and C are sets such that $A \oplus C = B \oplus C$. Must it be the case that $A = B$?

32. If A, B, C, and D are sets, does it follow that $(A \oplus B) \oplus (C \oplus D) = (A \oplus C) \oplus (B \oplus D)$?

33. If A, B, C, and D are sets, does it follow that $(A \oplus B) \oplus (C \oplus D) = (A \oplus D) \oplus (B \oplus C)$?

***34.** Show that if A, B, and C are sets, then

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|.$$

(This is a special case of the inclusion–exclusion principle, which will be studied in Chapter 5.)

35. Let $A_i = \{1, 2, 3, \ldots, i\}$ for $i = 1, 2, 3, \ldots$. Find

a) $\bigcup_{i=1}^{n} A_i$. **b)** $\bigcap_{i=1}^{n} A_i$.

36. Let $A_i = \{i, i+1, i+2, \ldots\}$. Find

a) $\bigcup_{i=1}^{n} A_i$. **b)** $\bigcap_{i=1}^{n} A_i$.

37. Let A_i be the set of all nonempty bit strings (that is, bit strings of length at least one) of length not exceeding i. Find

a) $\bigcup_{i=1}^{n} A_i$. **b)** $\bigcap_{i=1}^{n} A_i$.

38. Suppose that the universal set is $U = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. Express each of the following sets with bit strings where the ith bit in the string is 1 if i is in the set and 0 otherwise.

a) $\{3, 4, 5\}$ **b)** $\{1, 3, 6, 10\}$
c) $\{2, 3, 4, 7, 8, 9\}$

39. Using the same universal set as in the last problem, find the set specified by each of the following bit strings.

a) 11 1100 1111 **b)** 01 0111 1000
c) 10 0000 0001

40. What subsets of a finite universal set do the following bit strings represent?

a) the string with all zeros
b) the string with all ones

41. What is the bit string corresponding to the difference of two sets?

42. What is the bit string corresponding to the symmetric difference of two sets?

43. Show how bitwise operations on bit strings can be used to find the following combinations of $A = \{a, b, c, d, e\}$, $B = \{b, c, d, g, p, t, v\}$, $C = \{c, e, i, o, u, x, y, z\}$, and $D = \{d, e, h, i, n, o, t, u, x, y\}$.

a) $A \cup B$ **b)** $A \cap B$
c) $(A \cup D) \cap (B \cup C)$ **d)** $A \cup B \cup C \cup D$

44. How can the union and intersection of n sets that all are subsets of the universal set U be found using bit strings?

45. The **successor** of the set A is the set $A \cup \{A\}$. Find the successors of the following sets.

a) $\{1, 2, 3\}$ **b)** $\emptyset$ **c)** $\{\emptyset\}$ **d)** $\{\emptyset, \{\emptyset\}\}$

46. How many elements does the successor of a set with n elements have?

Sometimes the number of times that an element occurs in an unordered collection matters. **Multisets** are unordered collections of elements where an element can occur as a member more than once. The notation $\{m_1 \cdot a_1, m_2 \cdot a_2, \ldots, m_r \cdot a_r\}$ denotes the multiset with element a_1 occurring m_1 times, element a_2 occurring m_2 times, and so on. The numbers m_i, $i = 1, 2, \ldots, r$ are called the **multiplicities** of the elements a_i, $i = 1, 2, \ldots, r$.

Let P and Q be multisets. The **union** of the multisets P and Q is the multiset where the multiplicity of an element is the maximum of its multiplicities in P and Q. The **intersection** of P and Q is the multiset where the multiplicity of an element is the minimum of its multiplicities in P and Q. The **difference** of P and Q is the multiset where the multiplicity of an element is the multiplicity of the element in P less its multiplicity in Q unless this difference is negative, in which case the multiplicity is 0. The **sum** of P and Q is the multiset where the multiplicity of an element is the sum of multiplicities in P and Q. The union, intersection, and difference of P and Q are denoted by $P \cup Q$, $P \cap Q$, and $P - Q$, respectively (where these operations should not be confused with the analogous operations for sets). The sum of P and Q is denoted by $P + Q$.

47. Let A and B be the multisets $\{3 \cdot a, 2 \cdot b, 1 \cdot c\}$ and $\{2 \cdot a, 3 \cdot b, 4 \cdot d\}$, respectively. Find

a) $A \cup B$. **b)** $A \cap B$. **c)** $A - B$.
d) $B - A$. **e)** $A + B$.

48. Suppose that A is the multiset that has as its elements the types of computer equipment needed by one department of a university where the multiplicities are the number of pieces of each type needed, and B is the analogous multiset for a second department of the university. For instance, A could be the multiset {107 · personal computers, 44 · routers, 6 · servers} and B could be the multiset {14 · personal computers, 6 · routers, 2 · mainframes}.

a) What combination of A and B represents the equipment the university should buy assuming both departments use the same equipment?

b) What combination of A and B represents the equipment that will be used by both departments if both departments use the same equipment?

c) What combination of A and B represents the equipment that the second department uses, but the first department does not, if both departments use the same equipment?

d) What combination of A and B represents the equipment that the university should purchase if the departments do not share equipment?

Fuzzy sets are used in artificial intelligence. Each element in the universal set U has a **degree of membership,** which is a real number between 0 and 1 (including 0 and 1), in a fuzzy set S. The fuzzy set S is denoted by listing the elements with their degrees of membership (elements with 0 degree of membership are not listed). For instance, we write {0.6 Alice, 0.9 Brian, 0.4 Fred, 0.1 Oscar, 0.5 Rita} for the set F (of famous people) to indicate that Alice has a 0.6 degree of membership in F, Brian has a 0.9 degree of membership in F, Fred has a 0.4 degree of membership in F, Oscar has a 0.1 degree of membership in F, and Rita has a 0.5 degree of membership in F (so that Brian is the most famous and Oscar is the least famous of these people). Also suppose that R is the set of rich people with R = {0.4 Alice, 0.8 Brian, 0.2 Fred, 0.9 Oscar, 0.7 Rita}.

49. The **complement** of a fuzzy set S is the set $\bar{S}$, with the degree of the membership of an element in $\bar{S}$ equal to 1 minus the degree of membership of this element in S. Find $\bar{F}$ (the fuzzy set of people who are not famous) and $\bar{R}$ (the fuzzy set of people who are not rich).

50. The **union** of two fuzzy sets S and T is the fuzzy set $S \cup T$, where the degree of membership of an element in $S \cup T$ is the maximum of the degrees of membership of this element in S and in T. Find the fuzzy set $F \cup R$ of rich or famous people.

51. The **intersection** of two fuzzy sets S and T is the fuzzy set $S \cap T$, where the degree of membership of an element in $S \cap T$ is the minimum of the degrees of membership of this element in S and in T. Find the fuzzy set $F \cap R$ of rich and famous people.

1.6

Functions

INTRODUCTION

In many instances we assign to each element of a set a particular element of a second set (which may be the same as the first). For example, suppose that each student in a discrete mathematics class is assigned a letter grade from the set $\{A, B, C, D, F\}$. And suppose that the grades are A for Adams, C for Chou, B for Goodfriend, A for Rodriguez, and F for Stevens. This assignment of grades is illustrated in Figure 1.

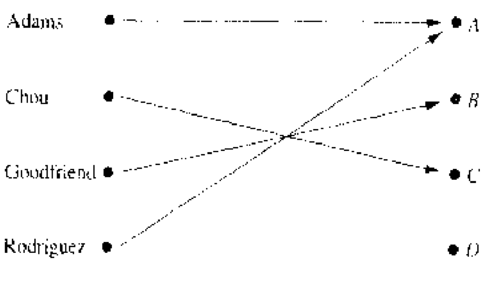


FIGURE 1 Assignment of Grades in a Discrete Mathematics Class.

This assignment is an example of a function. The concept of a function is extremely important in discrete mathematics. Functions are used in the definition of such discrete structures as sequences and strings. Functions are also used to represent how long it takes a computer to solve problems of a given size. Recursive functions, which are functions defined in terms of themselves, are used throughout computer science; they will be studied in Chapter 3. This section reviews the basic concepts involving functions needed in discrete mathematics.

DEFINITION 1. **Let A and B be sets. A *function* f from A to B is an assignment of exactly one element of B to each element of A. We write $f(a) = b$ if b is the unique element of B assigned by the function f to the element a of A. If f is a function from A to B, we write $f : A \to B$.**

Functions are specified in many different ways. Sometimes we explicitly state the assignments. Often we give a formula, such as $f(x) = x + 1$, to define a function. Other times we use a computer program to specify a function.

DEFINITION 2. **If f is a function from A to B, we say that A is the *domain* of f and B is the *codomain* of f. If $f(a) = b$, we say that b is the *image* of a and a is a *pre-image* of b. The *range* of f is the set of all images of elements of A. Also, if f is a function from A to B, we say that f *maps* A to B.**

Figure 2 represents a function f from A to B.

Consider the example that began this section. Let G be the function that assigns a grade to a student in our discrete mathematics class. Note that $G(\text{Adams}) = A$, for instance. The domain of G is the set {Adams, Chou, Goodfriend, Rodriguez, Stevens}, and the codomain is the set $\{A, B, C, D, F\}$. The range of G is the set $\{A, B, C, F\}$, because there are students who are assigned each grade except D. Also consider the following examples.

EXAMPLE 1 Let f be the function that assigns the last two bits of a bit string of length 2 or greater to that string. Then, the domain of f is the set of all bit strings of length 2 or greater, and both the codomain and range are the set {00, 01, 10, 11}. ■

EXAMPLE 2 Let f be the function from $\mathbf{Z}$ to $\mathbf{Z}$ that assigns the square of an integer to this integer. Then, $f(x) = x^2$, where the domain of f is the set of all integers, the codomain of f can be chosen to be the set of all integers, and the range of f is the set of all nonnegative integers that are perfect squares, namely, $\{0, 1, 4, 9, \ldots\}$. ■

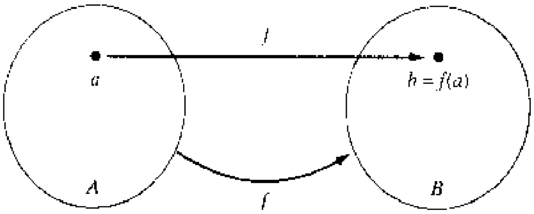


FIGURE 2 The Function f Maps A to B.

EXAMPLE 3 (For students familiar with Pascal) The domain and codomain of functions are often specified in programming languages. For instance, the Pascal statement

```
function floor(x: real): integer
```

states that the domain of the floor function is the set of real numbers and its codomain is the set of integers. ■

Two real-valued functions with the same domain can be added and multiplied.

DEFINITION 3. Let f_1 and f_2 be functions from A to $\mathbf{R}$. Then $f_1 + f_2$ and $f_1 f_2$ are also functions from A to $\mathbf{R}$ defined by

$$(f_1 + f_2)(x) = f_1(x) + f_2(x),$$
$$(f_1 f_2)(x) = f_1(x) f_2(x).$$

Note that the functions $f_1 + f_2$ and $f_1 f_2$ have been defined by specifying their values at x in terms of the values of f_1 and f_2 at x.

EXAMPLE 4 Let f_1 and f_2 be functions from $\mathbf{R}$ to $\mathbf{R}$ such that $f_1(x) = x^2$ and $f_2(x) = x - x^2$. What are the functions $f_1 + f_2$ and $f_1 f_2$?

Solution: From the definition of the sum and product of functions, it follows that

$$(f_1 + f_2)(x) = f_1(x) + f_2(x) = x^2 + (x - x^2) = x$$

and

$$(f_1 f_2)(x) = x^2(x - x^2) = x^3 - x^4.$$ ■

When f is a function from a set A to a set B, the image of a subset of A can also be defined.

DEFINITION 4. Let f be a function from the set A to the set B and let S be a subset of A. The *image* of S is the subset of B that consists of the images of the elements of S. We denote the image of S by $f(S)$, so that

$$f(S) = \{f(s) \mid s \in S\}.$$

EXAMPLE 5 Let $A = \{a, b, c, d, e\}$ and $B = \{1, 2, 3, 4\}$ with $f(a) = 2$, $f(b) = 1$, $f(c) = 4$, $f(d) = 1$, and $f(e) = 1$. The image of the subset $S = \{b, c, d\}$ is the set $f(S) = \{1, 4\}$. ■

ONE-TO-ONE AND ONTO FUNCTIONS

Some functions have distinct images at distinct members of their domain. These functions are said to be **one-to-one.**

DEFINITION 5. **A function f is said to be *one-to-one*, or *injective*, if and only if $f(x) = f(y)$ implies that $x = y$ for all x and y in the domain of f. A function is said to be an *injection* if it is one-to-one.**

Remark: A function f is one-to-one if and only if $f(x) \neq f(y)$ whenever $x \neq y$. This way of expressing that f is one-to-one is obtained by taking the contrapositive of the implication in the definition.

We illustrate this concept by giving examples of functions that are one-to-one and other functions that are not one-to-one.

EXAMPLE 6 Determine whether the function f from $\{a, b, c, d\}$ to $\{1, 2, 3, 4, 5\}$ with $f(a) = 4$, $f(b) = 5$, $f(c) = 1$, and $f(d) = 3$ is one-to-one.

Solution: The function f is one-to-one since f takes on different values at the four elements of its domain. This is illustrated in Figure 3. ■

EXAMPLE 7 Determine whether the function $f(x) = x^2$ from the set of integers to the set of integers is one-to-one.

Solution: The function $f(x) = x^2$ is not one-to-one because, for instance, $f(1) = f(-1) = 1$, but $1 \neq -1$. ■

EXAMPLE 8 Determine whether the function $f(x) = x + 1$ is one-to-one.

Solution: The function $f(x) = x + 1$ is a one-to-one function. To demonstrate this, note that $x + 1 \neq y + 1$ when $x \neq y$. ■

We now give some conditions that guarantee that a function is one-to-one.

DEFINITION 6. **A function f whose domain and codomain are subsets of the set of real numbers is called *strictly increasing* if $f(x) < f(y)$ whenever $x < y$ and x and y are in the domain of f. Similarly, f is called *strictly decreasing* if $f(x) > f(y)$ whenever $x < y$ and x and y are in the domain of f.**

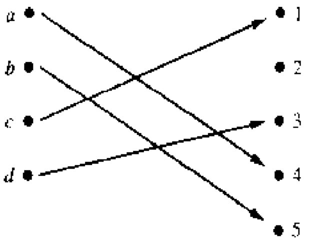


FIGURE 3 A One-to-One Function.

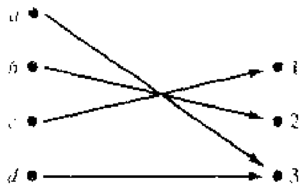


FIGURE 4 An Onto Function.

From these definitions, we see that a function that is either strictly increasing or strictly decreasing must be one-to-one.

For some functions the range and the codomain are equal. That is, every member of the codomain is the image of some element of the domain. Functions with this property are called **onto** functions.

DEFINITION 7. **A function f from A to B is called *onto*, or *surjective*, if and only if for every element $b \in B$ there is an element $a \in A$ with $f(a) = b$. A function f is called a *surjection* if it is onto.**

We now give examples of onto functions and functions that are not onto.

EXAMPLE 9 Let f be the function from $\{a, b, c, d\}$ to $\{1, 2, 3\}$ defined by $f(a) = 3$, $f(b) = 2$, $f(c) = 1$, and $f(d) = 3$. Is f an onto function?

Solution: Since all three elements of the codomain are images of elements in the domain, we see that f is onto. This is illustrated in Figure 4. ■

EXAMPLE 10 Is the function $f(x) = x^2$ from the set of integers to the set of integers onto?

Solution: The function f is not onto since there is no integer x with $x^2 = -1$, for instance. ■

EXAMPLE 11 Is the function $f(x) = x + 1$ from the set of integers to the set of integers onto?

Solution: This function is onto, since for every integer y there is an integer x such that $f(x) = y$. To see this, note that $f(x) = y$ if and only if $x + 1 = y$, which holds if and only if $x = y - 1$. ■

DEFINITION 8. **The function f is a *one-to-one correspondence*, or a *bijection*, if it is both one-to-one and onto.**

The following examples illustrate the concept of a bijection.

EXAMPLE 12 Let f be the function from $\{a, b, c, d\}$ to $\{1, 2, 3, 4\}$ with $f(a) = 4$, $f(b) = 2$, $f(c) = 1$, and $f(d) = 3$. Is f a bijection?

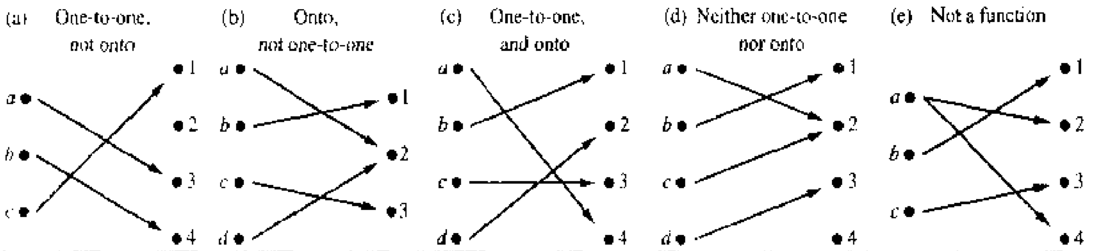


FIGURE 5 **Examples of Different Types of Correspondences.**

Solution: The function f is one-to-one and onto. It is one-to-one since the function takes on distinct values. It is onto since all four elements of the codomain are images of elements in the domain. Hence, f is a bijection. ■

Figure 5 displays four functions where the first is one-to-one but not onto, the second is onto but not one-to-one, the third is both one-to-one and onto, and the fourth is neither one-to-one nor onto. The fifth correspondence in Figure 5 is not a function, since it sends an element to two different elements.

Suppose that f is a function from a set A to itself. If A is finite, then f is one-to-one if and only if it is onto. (This follows from the result in Exercise 58 at the end of this section.) This is not necessarily the case if A is infinite (as will be shown in Section 1.7).

EXAMPLE 13 Let A be a set. The *identity function* on A is the function $\iota_A : A \rightarrow A$ where

$$\iota_A(x) = x$$

where $x \in A$. In other words, the identity function ι_A is the function that assigns each element to itself. The function ι_A is one-to-one and onto, so that it is a bijection. ■

INVERSE FUNCTIONS AND COMPOSITIONS OF FUNCTIONS

Now consider a one-to-one correspondence f from the set A to the set B. Since f is an onto function, every element of B is the image of some element in A. Furthermore, because f is also a one-to-one function, every element of B is the image of a *unique* element of A. Consequently, we can define a new function from B to A that reverses the correspondence given by f. This leads to the following definition.

DEFINITION 9. **Let f be a one-to-one correspondence from the set A to the set B. The *inverse function* of f is the function that assigns to an element b belonging to B the unique element a in A such that $f(a) = b$. The inverse function of f is denoted by f^{-1}. Hence, $f^{-1}(b) = a$ when $f(a) = b$.**

Figure 6 illustrates the concept of a an inverse function.

If a function f is not a one-to-one correspondence, we cannot define an inverse function of f. When f is not a one-to-one correspondence, either it is not one-to-one or it is not onto. If f is not one-to-one, some element b in the codomain is the image of more than one element in the domain. If f is not onto, for some element b in the codomain, no element a in the domain exists for which $f(a) = b$. Consequently, if f is not a

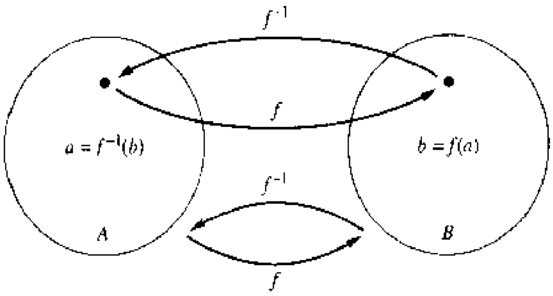


FIGURE 6 **The Function f^{-1} Is the Inverse of Function f.**

one-to-one correspondence, we cannot assign to each element b in the codomain a unique element a in the domain such that $f(a) = b$ (because for some b there is either more than one such a or no such a).

A one-to-one correspondence is called **invertible** since we can define an inverse of this function. A function is **not invertible** if it is not a one-to-one correspondence, since the inverse of such a function does not exist.

EXAMPLE 14 Let f be the function from $\{a, b, c\}$ to $\{1, 2, 3\}$ such that $f(a) = 2$, $f(b) = 3$, and $f(c) = 1$. Is f invertible, and if it is, what is its inverse?

Solution: The function f is invertible since it is a one-to-one correspondence. The inverse function f^{-1} reverses the correspondence given by f, so that $f^{-1}(1) = c$, $f^{-1}(2) = a$, and $f^{-1}(3) = b$. ■

EXAMPLE 15 Let f be the function from the set of integers to the set of integers such that $f(x) = x + 1$. Is f invertible, and if it is, what is its inverse?

Solution: The function f has an inverse since it is a one-to-one correspondence, as we have shown. To reverse the correspondence, suppose that y is the image of x, so that $y = x + 1$. Then $x = y - 1$. This means that $y - 1$ is the unique element of **Z** that is sent to y by f. Consequently, $f^{-1}(y) = y - 1$. ■

EXAMPLE 16 Let f be the function from **Z** to **Z** with $f(x) = x^2$. Is f invertible?

Solution: Since $f(-1) = f(1) = 1$, f is not one-to-one. If an inverse function were defined, it would have to assign two elements to 1. Hence, f is not invertible. ■

DEFINITION 10. Let g be a function from the set A to the set B and let f be a function from the set B to the set C. The *composition* of the functions f and g, denoted by $f \circ g$, is defined by

$$(f \circ g)(a) = f(g(a)).$$

In other words, $f \circ g$ is the function that assigns to the element a of A the element assigned by f to $g(a)$. Note that the composition $f \circ g$ cannot be defined unless the range of g is a subset of the domain of f. In Figure 7 the composition of functions is shown.

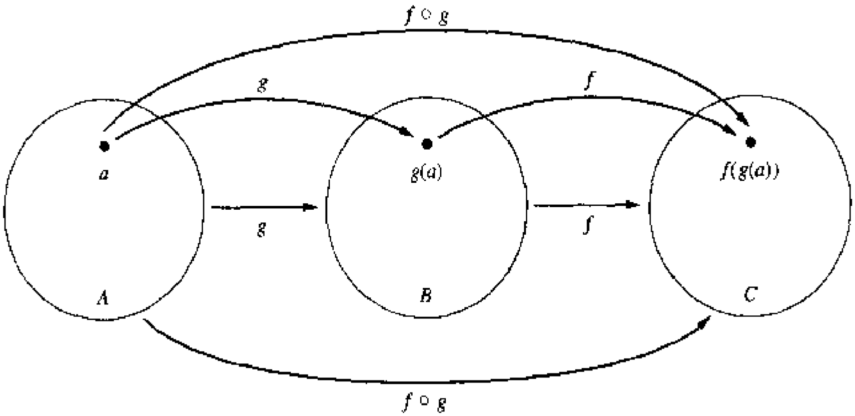


FIGURE 7 **The Composition of the Functions f and g.**

EXAMPLE 17 Let g be the function from the set $\{a, b, c\}$ to itself such that $g(a) = b$, $g(b) = c$, and $g(c) = a$. Let f be the function from the set $\{a, b, c\}$ to the set $\{1, 2, 3\}$ such that $f(a) = 3$, $f(b) = 2$, and $f(c) = 1$. What is the composition of f and g, and what is the composition of g and f?

Solution: The composition $f \circ g$ is defined by $(f \circ g)(a) = f(g(a)) = f(b) = 2$, $(f \circ g)(b) = f(g(b)) = f(c) = 1$, and $(f \circ g)(c) = f(g(c)) = f(a) = 3$.

Note that $g \circ f$ is not defined, because the range of f is not a subset of the domain of g. ∎

EXAMPLE 18 Let f and g be the functions from the set of integers to the set of integers defined by $f(x) = 2x + 3$ and $g(x) = 3x + 2$. What is the composition of f and g? What is the composition of g and f?

Solution: Both the compositions $f \circ g$ and $g \circ f$ are defined. Moreover,

$$(f \circ g)(x) = f(g(x)) = f(3x + 2) = 2(3x + 2) + 3 = 6x + 7$$

and

$$(g \circ f)(x) = g(f(x)) = g(2x + 3) = 3(2x + 3) + 2 = 6x + 11$$ ∎

Remark: Note that even though $f \circ g$ and $g \circ f$ are defined for the functions f and g in Example 18, $f \circ g$ and $g \circ f$ are not equal. In other words, the commutative law does not hold for the composition of functions.

When the composition of a function and its inverse is formed, in either order, an identity function is obtained. To see this, suppose that f is a one-to-one correspondence from the set A to the set B. Then the inverse function f^{-1} exists and is a one-to-one correspondence from B to A. The inverse function reverses the correspondence of the original function, so that $f^{-1}(b) = a$ when $f(a) = b$, and $f(a) = b$ when $f^{-1}(b) = a$. Hence,

$$(f^{-1} \circ f)(a) = f^{-1}(f(a)) = f^{-1}(b) = a,$$

and

$$(f \circ f^{-1})(b) = f(f^{-1}(b)) = f(a) = b.$$

Consequently $f^{-1} \circ f = \iota_A$ and $f \circ f^{-1} = \iota_B$, where ι_A and ι_B are the identity functions on the sets A and B, respectively. That is, $(f^{-1})^{-1} = f$.

THE GRAPHS OF FUNCTIONS

We can associate a set of pairs in $A \times B$ to each function from A to B. This set of pairs is called the **graph** of the function and is often displayed pictorially to aid in understanding the behavior of the function.

DEFINITION 11. Let f be a function from the set A to the set B. The *graph* of the function f is the set of ordered pairs $\{(a, b) \mid a \in A$ and $f(a) = b\}$.

From the definition, the graph of a function f from A to B is the subset of $A \times B$ containing the ordered pairs with the second entry equal to the element of B assigned by f to the first entry.

EXAMPLE 19 Display the graph of the function $f(n) = 2n + 1$ from the set of integers to the set of integers.

Solution: The graph of f is the set of ordered pairs of the form $(n, 2n + 1)$ where n is an integer. This graph is displayed in Figure 8. ■

EXAMPLE 20 Display the graph of the function $f(x) = x^2$ from the set of integers to the set of integers.

Solution: The graph of f is the set of ordered pairs of the form $(x, f(x)) = (x, x^2)$ where x is an integer. This graph is displayed in Figure 9. ■

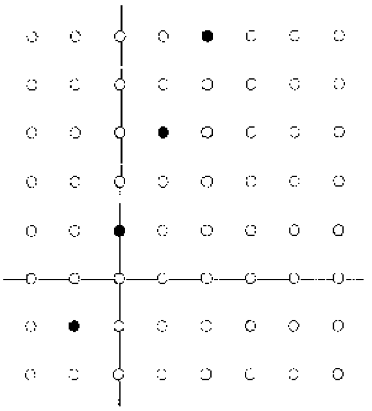


FIGURE 8 The Graph of the Function $f(n) = 2n + 1$ from Z to Z.

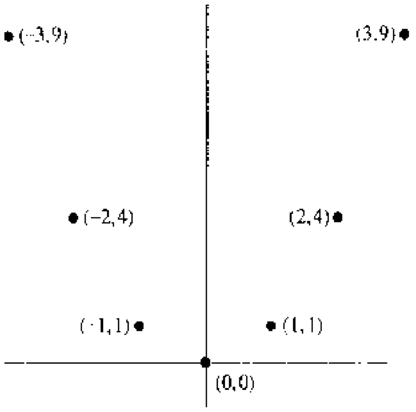


FIGURE 9 The Graph of $f(x) = x^2$ from Z to Z.

SOME IMPORTANT FUNCTIONS

Next, we introduce two important functions in discrete mathematics, namely, the floor and ceiling functions. Let x be a real number. The floor function rounds x down to the closest integer less than or equal to x, and the ceiling function rounds x up to the closest integer greater than or equal to x. These functions are often used when objects are counted. They play an important role in the analysis of the number of steps used by procedures to solve problems of a particular size.

DEFINITION 12. The *floor function* assigns to the real number x the largest integer that is less than or equal to x. The value of the floor function at x is denoted by $\lfloor x \rfloor$. The *ceiling function* assigns to the real number x the smallest integer that is greater than or equal to x. The value of the ceiling function at x is denoted by $\lceil x \rceil$.

Remark: The floor function is often also called the *greatest integer function*. It is often denoted by $[x]$.

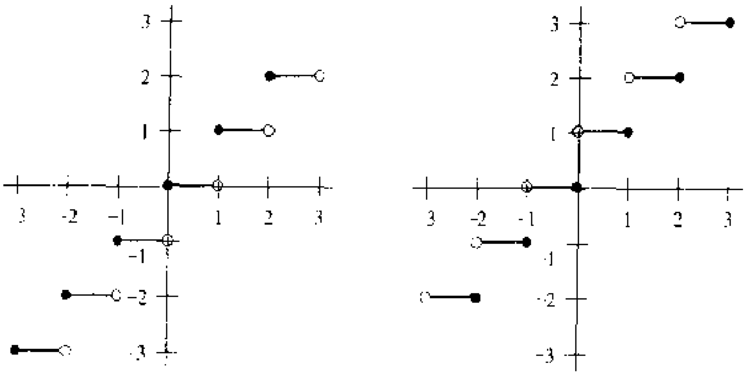
EXAMPLE 21 The following are some values of the floor and ceiling functions:

$$\lfloor \tfrac{1}{2} \rfloor = 0, \quad \lceil \tfrac{1}{2} \rceil = 1, \quad \lfloor -\tfrac{1}{2} \rfloor = -1, \quad \lceil -\tfrac{1}{2} \rceil = 0,$$
$$\lfloor 3.1 \rfloor = 3, \quad \lceil 3.1 \rceil = 4, \quad \lfloor 7 \rfloor = 7, \quad \lceil 7 \rceil = 7.$$

■

We display the graphs of the floor and ceiling functions in Figure 10.

The floor and ceiling functions are useful in a wide variety of applications, including those involving data storage and data transmission. Consider the following examples, typical of basic calculations done when database and data communications problems are studied.



(a) $y = \lfloor x \rfloor$ (b) $y = \lceil x \rceil$

FIGURE 10 **Graphs of the (a) Floor and (b) Ceiling Functions.**

EXAMPLE 22 Data stored on a computer disk or transmitted over a data network are usually represented as a string of bytes. Each byte is made up of 8 bits. How many bytes are required to encode 100 bits of data?

Solution: To determine the number of bytes needed, we determine the smallest integer that is at least as large as the quotient when 100 is divided by 8, the number of bits in a byte. Consequently, $\lceil 100/8 \rceil = \lceil 12.5 \rceil = 13$ bytes is required. ■

EXAMPLE 23 In asynchronous transfer mode (ATM) (a communications protocol used on backbone networks), data are organized into cells of 53 bytes. How many ATM cells can be transmitted in 1 minute over a connection that transmits data at the rate of 500 kilobits per second?

Solution: In 1 minute, this connection can transmit $500{,}000 \cdot 60 = 30{,}000{,}000$ bits. Each ATM cell is 53 bytes long, which means that it is $53 \cdot 8 = 424$ bits long. To determine the number of cells that can be transmitted in 1 minute, we determine the largest integer not exceeding the quotient when 30,000,000 is divided by 424. Consequently, $\lfloor 30{,}000{,}000/424 \rfloor = 70{,}754$ ATM cells can be transmitted in 1 minute over a 500 kilobit per second connection. ■

Table 1, with x denoting a real number, displays some simple but important properties of the floor and ceiling functions. Since these functions appear so frequently in discrete mathematics, it is useful to look over these identities. Each property in this table can be established using the definitions of the floor and ceiling functions. Properties (1a), (1b), (1c), and (1d) follow directly from these definitions. For example, (1a) states that $\lfloor x \rfloor = n$ if and only if the integer n is less than or equal to x and $n + 1$ is larger than x. This is precisely what it means for n to be the greatest integer not exceeding x, which is the definition of $\lfloor x \rfloor = n$. Properties (1b), (1c), and (1d) can be established similarly.

We will show that (4a) is true. To show that (4a) is true, suppose that $\lfloor x \rfloor = n$ where n is an integer. By (1a) it follows that $n \le x < n + 1$. Adding m to this inequality shows that $n+m \le x+m < n+m+1$. Using (1a) again, we see that $\lfloor x+m \rfloor = n+m = \lfloor x \rfloor + m$,

TABLE 1 Useful Properties of the Floor and Ceiling Functions.
(1a) $\lfloor x \rfloor = n$ if and only if $n \le x < n + 1$ where n is an integer (1b) $\lceil x \rceil = n$ if and only if $n - 1 < x \le n$ where n is an integer (1c) $\lfloor x \rfloor = n$ if and only if $x - 1 < n \le x$ where n is an integer (1d) $\lceil x \rceil = n$ if and only if $x \le n < x + 1$ where n is an integer
(2) $x - 1 < \lfloor x \rfloor \le x \le \lceil x \rceil < x + 1$
(3a) $\lfloor -x \rfloor = -\lceil x \rceil$ (3b) $\lceil -x \rceil = -\lfloor x \rfloor$
(4a) $\lfloor x + m \rfloor = \lfloor x \rfloor + m$ when m is an integer (4b) $\lceil x + m \rceil = \lceil x \rceil + m$ when m is an integer

which is what we wanted to show. We defer establishing the other properties to the exercises.

There are certain types of functions that will be used throughout the text. These include polynomial, logarithmic, and exponential functions. A brief review of the properties of these functions needed in this text is given in Appendix 1. In this book the notation $\log x$ will be used to denote the logarithm to the base 2 of x, since 2 is the base that we will usually use for logarithms. We will denote logarithms to the base b, where b is any real number greater than 1, by $\log_b x$.

Exercises

1. Why is f not a function from **R** to **R** in the following equations?
  a) $f(x) = 1/x$ **b)** $f(x) = \sqrt{x}$
  c) $f(x) = \pm\sqrt{(x^2 + 1)}$

2. Determine whether f is a function from **Z** to **R** if
  a) $f(n) = \pm n$. **b)** $f(n) = \sqrt{n^2 + 1}$.
  c) $f(n) = 1/(n^2 - 4)$.

3. Determine whether f is a function from the set of all bit strings to the set of integers if
  a) $f(S)$ is the position of a 0 bit in S.
  b) $f(S)$ is the number of 1 bits in S.
  c) $f(S)$ is the smallest integer i such that the ith bit of S is 1 and $f(S) = 0$ when S is the empty string, the string with no bits.

4. *Find the domain and range of the following functions.*
  a) the function that assigns to each nonnegative integer its last digit
  b) the function that assigns the next largest integer to a positive integer
  c) the function that assigns to a bit string the number of one bits in the string
  d) the function that assigns to a bit string the number of bits in the string

5. *Find the domain and range of the following functions.*
  a) the function that assigns to each bit string the difference between the number of ones and the number of zeros
  b) the function that assigns to each bit string twice the number of zeros in that string
  c) the function that assigns the number of bits left over when a bit string is split into bytes (which are blocks of 8 bits)
  d) *the function that assigns to each positive integer the largest perfect square not exceeding this integer*

6. Find the following values.
  a) $\lfloor 1.1 \rfloor$ **b)** $\lceil 1.1 \rceil$ **c)** $\lfloor -0.1 \rfloor$
  d) $\lceil -0.1 \rceil$ **e)** $\lceil 2.99 \rceil$ **f)** $\lceil -2.99 \rceil$
  g) $\lfloor \frac{1}{2} + \lceil \frac{1}{2} \rceil \rfloor$ **h)** $\lceil \lfloor \frac{1}{2} \rfloor + \lceil \frac{1}{2} \rceil + \frac{1}{2} \rceil$

7. Find the following values.
  a) $\lceil \frac{3}{4} \rceil$ **b)** $\lfloor \frac{7}{8} \rfloor$ **c)** $\lceil -\frac{3}{4} \rceil$ **d)** $\lfloor -\frac{7}{8} \rfloor$
  e) $\lceil 3 \rceil$ **f)** $\lfloor -1 \rfloor$ **g)** $\lfloor \frac{1}{2} + \lceil \frac{3}{2} \rceil \rfloor$ **h)** $\lfloor \frac{1}{2} \cdot \lfloor \frac{5}{2} \rfloor \rfloor$

8. Determine whether each of the following functions from $\{a, b, c, d\}$ to itself is one-to-one.
  a) $f(a) = b, f(b) = a, f(c) = c, f(d) = d$
  b) $f(a) = b, f(b) = b, f(c) = d, f(d) = c$
  c) $f(a) = d, f(b) = b, f(c) = c, f(d) = d$

9. Which functions in Exercise 8 are onto?

10. Determine whether each of the following functions from **Z** to **Z** is one-to-one.
  a) $f(n) = n - 1$ **b)** $f(n) = n^2 + 1$
  c) $f(n) = n^3$ **d)** $f(n) = \lceil n/2 \rceil$

11. Which functions in Exercise 10 are onto?

12. Give an example of a function from **N** to **N** that is
  a) one-to-one but not onto.
  b) onto but not one-to-one.
  c) both onto and one-to-one (but different from the identity function).
  d) neither one-to-one nor onto.

13. Give an explicit formula for a function from the set of integers to the set of positive integers that is
  a) one-to-one, but not onto.
  b) onto, but not one-to-one.
  c) one-to-one and onto.
  d) neither one-to-one nor onto.

14. Determine whether each of the following functions is a bijection from **R** to **R**.
  a) $f(x) = -3x + 4$
  b) $f(x) = -3x^2 + 7$
  c) $f(x) = (x + 1)/(x + 2)$
  d) $f(x) = x^5 + 1$

15. Determine whether each of the following functions is a bijection from **R** to **R**.
  a) $f(x) = 2x + 1$
  b) $f(x) = x^2 + 1$
  c) $f(x) = x^3$
  d) $f(x) = (x^2 + 1)/(x^2 + 2)$

16. Let $S = \{-1, 0, 2, 4, 7\}$. Find $f(S)$ if
  a) $f(x) = 1$. **b)** $f(x) = 2x + 1$.
  c) $f(x) = \lceil x/5 \rceil$. **d)** $f(x) = \lfloor (x^2 + 1)/3 \rfloor$.

17. Let $f(x) = \lfloor x^2/3 \rfloor$. Find $f(S)$ if
  a) $S = \{-2, -1, 0, 1, 2, 3\}$. **b)** $S = \{0, 1, 2, 3, 4, 5\}$.
  c) $S = \{1, 5, 7, 11\}$. **d)** $S = \{2, 6, 10, 14\}$.

18. Let $f(x) = 2x$. What is
  a) $f(\mathbf{Z})$? **b)** $f(\mathbf{N})$? **c)** $f(\mathbf{R})$?

19. Suppose that g is a function from A to B and f is a function from B to C.
  a) Show that if both f and g are one-to-one functions, then $f \circ g$ is also one-to-one.
  b) Show that if both f and g are onto functions, then $f \circ g$ is also onto.

***20.** If f and $f \circ g$ are one-to-one, does it follow that g is one-to-one? Justify your answer.

***21.** If f and $f \circ g$ are onto, does it follow that g is onto? Justify your answer.

22. Find $f \circ g$ and $g \circ f$ where $f(x) = x^2 + 1$ and $g(x) = x + 2$ are functions from $\mathbf{R}$ to $\mathbf{R}$.

23. Find $f + g$ and fg for the functions f and g given in Exercise 22.

24. Let $f(x) = ax + b$ and $g(x) = cx + d$ where a, b, c, and d are constants. Determine for which constants a, b, c, and d it is true that $f \circ g = g \circ f$.

25. Show that the function $f(x) = ax + b$ from $\mathbf{R}$ to $\mathbf{R}$ is invertible, where a and b are constants, with $a \neq 0$, and find the inverse of f.

26. Let f be a function from the set A to the set B. Let S and T be subsets of A. Show that
  a) $f(S \cup T) = f(S) \cup f(T)$.
  b) $f(S \cap T) \subseteq f(S) \cap f(T)$.

27. Give an example to show that the inclusion in part (b) in Exercise 26 may be proper.

Let f be a function from the set A to the set B. Let S be a subset of B. We define the **inverse image** of S to be the subset of A containing all pre-images of all elements of S. We denote the inverse image of S by $f^{-1}(S)$, so that $f^{-1}(S) = \{a \in A \mid f(a) \in S\}$.

28. Let f be the function from $\mathbf{R}$ to $\mathbf{R}$ defined by $f(x) = x^2$. Find
  a) $f^{-1}(\{1\})$. **b)** $f^{-1}(\{x \mid 0 < x < 1\})$.
  c) $f^{-1}(\{x \mid x > 4\})$.

29. Let $g(x) = \lfloor x \rfloor$. Find
  a) $g^{-1}(\{0\})$. **b)** $g^{-1}(\{-1, 0, 1\})$.
  c) $g^{-1}(\{x \mid 0 < x < 1\})$.

30. Let f be a function from A to B. Let S and T be subsets of B. Show that
  a) $f^{-1}(S \cup T) = f^{-1}(S) \cup f^{-1}(T)$.
  b) $f^{-1}(S \cap T) = f^{-1}(S) \cap f^{-1}(T)$.

31. Let f be a function from A to B. Let S be a subset of B. Show that $\overline{f^{-1}(S)} = f^{-1}(\overline{S})$.

32. Show that $\lfloor x + \frac{1}{2} \rfloor$ is the closest integer to the integer x, except when x is midway between two integers, when it is the larger of these two integers.

33. Show that $\lceil x - \frac{1}{2} \rceil$ is the closest integer to the integer x, except when x is midway between two integers, when it is the smaller of these two integers.

34. Show that if x is a real number, then $\lceil x \rceil - \lfloor x \rfloor = 1$ if x is not an integer and $\lceil x \rceil - \lfloor x \rfloor = 0$ if x is an integer.

35. Show that if x is a real number, then $x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1$.

36. Show that if x is a real number and m is an integer, then $\lceil x + m \rceil = \lceil x \rceil + m$.

37. Show that if x is a real number and n is an integer, then
  a) $x < n$ if and only if $\lfloor x \rfloor < n$.
  b) $n < x$ if and only if $n < \lceil x \rceil$.

38. Show that if x is a real number and n is an integer, then
  a) $x \leq n$ if and only if $\lceil x \rceil \leq n$.
  b) $n \leq x$ if and only if $n \leq \lfloor x \rfloor$.

39. Let x be a real number. Show that $\lfloor 2x \rfloor = \lfloor x \rfloor + \lfloor x + \frac{1}{2} \rfloor$.

40. Prove that if x is a real number, then $\lfloor -x \rfloor = -\lceil x \rceil$ and $\lceil -x \rceil = -\lfloor x \rfloor$.

41. The function INT is found on some calculators, where $\mathrm{INT}(x) = \lfloor x \rfloor$ when x is a nonnegative real number and $\mathrm{INT}(x) = \lceil x \rceil$ when x is a negative real number. Show that this INT function satisfies the identity $\mathrm{INT}(-x) = -\mathrm{INT}(x)$.

42. Let a and b be real numbers with $a < b$. Use the floor and/or ceiling functions to express the number of integers n that satisfy the inequality $a \leq n \leq b$.

43. Let a and b be real numbers with $a < b$. Use the floor and/or ceiling functions to express the number of integers n that satisfy the inequality $a < n < b$.

44. How many bytes are required to encode n bits of data where n equals
  a) 4? **b)** 10? **c)** 500? **d)** 3000?

45. How many bytes are required to encode n bits of data where n equals
  a) 7? **b)** 17? **c)** 1001? **d)** 28800?

46. How many ATM cells (described in Example 23) can be transmitted in 10 seconds over a link operating at the following rates?
  a) 128 kilobits per second (1 kilobit = 1000 bits)
  b) 300 kilobits per second
  c) 1 megabit per second (1 megabit = 1,000,000 bits)

47. Data are transmitted over a particular Ethernet network in blocks of 1500 octets (blocks of 8 bits). How many blocks are required to transmit the following amounts of data over this Ethernet network? (Note that a byte is a synonym for an octet, a kilobyte is 1000 bytes, and a megabyte is 1,000,000 bytes.)
  a) 150 kilobytes of data
  b) 384 kilobytes of data
  c) 1.544 megabytes of data
  d) 45.3 megabytes of data

48. Draw the graph of the function $f(n) = 1 - n^2$ from $\mathbf{Z}$ to $\mathbf{Z}$.

49. Draw the graph of the function $f(x) = \lfloor 2x \rfloor$ from $\mathbf{R}$ to $\mathbf{R}$.

50. Draw the graph of the function $f(x) = \lfloor x/2 \rfloor$ from $\mathbf{R}$ to $\mathbf{R}$.

51. Draw the graph of the function $f(x) = \lfloor x \rfloor + \lfloor x/2 \rfloor$ from $\mathbf{R}$ to $\mathbf{R}$.

52. Draw the graph of the function $f(x) = \lceil x \rceil + \lfloor x/2 \rfloor$ from $\mathbf{R}$ to $\mathbf{R}$.

53. Draw graphs of each of the following functions.
- **a)** $f(x) = \lfloor x + \frac{1}{2} \rfloor$
- **b)** $f(x) = \lfloor 2x + 1 \rfloor$
- **c)** $f(x) = \lceil x/3 \rceil$
- **d)** $f(x) = \lceil 1/x \rceil$
- **e)** $f(x) = \lceil x - 2 \rceil + \lfloor x + 2 \rfloor$
- **f)** $f(x) = \lfloor 2x \rfloor \lceil x/2 \rceil$
- **g)** $f(x) = \lceil \lfloor x - \frac{1}{2} \rfloor + \frac{1}{2} \rceil$

54. Draw graphs of each of the following functions.
- **a)** $f(x) = \lceil 3x - 2 \rceil$
- **b)** $f(x) = \lceil 0.2x \rceil$
- **c)** $f(x) = \lfloor -1/x \rfloor$
- **d)** $f(x) = \lfloor x^2 \rfloor$
- **e)** $f(x) = \lceil x/2 \rceil \lfloor x/2 \rfloor$
- **f)** $f(x) = \lfloor x/2 \rfloor + \lceil x/2 \rceil$
- **g)** $f(x) = \lfloor 2 \lceil x/2 \rceil + \frac{1}{2} \rfloor$

55. Find the inverse function of $f(x) = x^3 + 1$.

56. Suppose that f is an invertible function from Y to Z and g is an invertible function from X to Y. Show that the inverse of the composition $f \circ g$ is given by $(f \circ g)^{-1} = g^{-1} \circ f^{-1}$.

57. Let S be a subset of a universal set U. The **characteristic function** f_S of S is the function from U to the set $\{0, 1\}$ such that $f_S(x) = 1$ if x belongs to S and $f_S(x) = 0$ if x does not belong to S. Let A and B be sets. Show that for all x
- **a)** $f_{A \cap B}(x) = f_A(x) \cdot f_B(x)$
- **b)** $f_{A \cup B}(x) = f_A(x) + f_B(x) - f_A(x) \cdot f_B(x)$
- **c)** $f_{\overline{A}}(x) = 1 - f_A(x)$
- **d)** $f_{A \oplus B}(x) = f_A(x) + f_B(x) - 2 f_A(x) f_B(x)$

58. Suppose that f is a function from A to B, where A and B are finite sets with $|A| = |B|$. Show that f is one-to-one if and only if it is onto.

A program designed to evaluate a function may not produce the correct value of the function for all elements in the domain of this function. For example, a program may not produce a correct value because evaluating the function may lead to an infinite loop or an overflow.

To study such situations, we use the concept of a partial function. A **partial function** f from a set A to a set B is an assignment to each element a in a subset of A, called the **domain of definition** of f, of a unique element b in B. The sets A and B are called the **domain** and **codomain** of f, respectively. We say that f is **undefined** for elements in A that are not in the domain of definition of f. We write $f : A \rightarrow B$ to denote that f is a partial function from A to B. (This is the same notation as is used for functions. The context in which the notation is used determines whether f is a partial function or a total function.) When the domain of definition of f equals A, we say that f is a **total function.**

59. For each of the following partial functions, determine its domain, codomain, domain of definition, and the set of values for which it is undefined. Also, determine whether it is a total function.
- **a)** $f : \mathbf{Z} \rightarrow \mathbf{R}$, $f(n) = 1/n$
- **b)** $f : \mathbf{Z} \rightarrow \mathbf{Z}$, $f(n) = \lceil n/2 \rceil$
- **c)** $f : \mathbf{Z} \times \mathbf{Z} \rightarrow \mathbf{Q}$, $f(m, n) = m/n$
- **d)** $f : \mathbf{Z} \times \mathbf{Z} \rightarrow \mathbf{Z}$, $f(m, n) = mn$
- **e)** $f : \mathbf{Z} \times \mathbf{Z} \rightarrow \mathbf{Z}$, $f(m, n) = m - n$ if $m > n$

60. **a)** Show that a partial function from A to B can be viewed as a function f^* from A to $B \cup \{u\}$ where u is not an element of B and

$$f^*(a) = \begin{cases} f(a) & \text{if } a \text{ belongs to the domain of definition of } f \\ u & \text{if } f \text{ is undefined at } a \end{cases}$$

b) Using the construction in (a), find the function f^* corresponding to each partial function in Exercise 59.

1.7

Sequences and Summations

INTRODUCTION

Sequences are used to represent ordered lists of elements. Sequences are used in discrete mathematics in many ways. They can be used to represent solutions to certain counting problems, as we will see in Chapter 5. They are also an important data structure in computer science. This section contains a review of the concept of a function, as well as the notation used to represent sequences and sums of terms of sequences.

When the elements of an infinite set can be listed, the set is called countable. We will conclude this section with a discussion of both countable and uncountable sets.

SEQUENCES

A sequence is a discrete structure used to represent an ordered list.

DEFINITION 1. A *sequence* is a function from a subset of the set of integers (usually either the set $\{0, 1, 2, \ldots\}$ or the set $\{1, 2, 3, \ldots\}$) to a set S. We use the notation a_n to denote the image of the integer n. We call a_n a *term* of the sequence.

We use the notation $\{a_n\}$ to describe the sequence. (Note that a_n represents an individual term of the sequence $\{a_n\}$. Also note that the notation $\{a_n\}$ for a sequence conflicts with the notation for a set. However, the context in which we use this notation will always make it clear when we are dealing with sets and when we are dealing with sequences.)

We describe sequences by listing the terms of the sequence in order of increasing subscripts.

EXAMPLE 1 Consider the sequence $\{a_n\}$, where

$$a_n = 1/n.$$

The list of the terms of this sequence, beginning with a_1, namely,

$$a_1, a_2, a_3, a_4, \ldots,$$

starts with

$$1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots.$$

■

EXAMPLE 2 Consider the sequence $\{b_n\}$ with $b_n = (-1)^n$. The list of the terms of this sequence, $b_0, b_1, b_2, b_3, \ldots$, begins with

$$1, -1, 1, -1, 1, \ldots.$$

■

EXAMPLE 3 Consider the sequence $c_n = 5^n$. The list of the terms of the sequence $c_0, c_1, c_2, c_3, c_4, c_5, \ldots$ begins with

$$1, 5, 25, 125, 625, 3125, \ldots.$$

■

Sequences of the form

$$a_1, a_2, \ldots, a_n$$

are often used in computer science. These finite sequences are also called **strings.** This string is also denoted by $a_1 a_2 \cdots a_n$. (Recall that bit strings, which are finite sequences of bits, were introduced in Section 1.1.) The **length** of the string S is the number of terms in this string. The **empty string** is the string that has no terms. The empty string has length zero.

EXAMPLE 4 The string *abcd* is a string of length four. ■

SPECIAL INTEGER SEQUENCES

A common problem in discrete mathematics is finding a formula or a general rule for constructing the terms of a sequence. Sometimes only a few terms of a sequence solving a problem are known; the goal is to identify the sequence. Even though the initial terms of a sequence do not determine the entire sequence (after all, there are infinitely many different sequences that start with any finite set of initial terms), knowing the first few terms may help you make an educated conjecture about the identity of your sequence. Once you have made this conjecture, you can try to verify that you have the correct sequence.

When trying to deduce a possible formula or rule for the terms of a sequence from the initial terms, try to find a pattern in these terms. You might also see whether you can determine how a term might have been produced from those preceding it. There are many questions you could ask, but some of the more useful are:

- Are there runs of the same value?
- Are terms obtained from previous terms by adding the same amount or an amount that depends on the position in the sequence?
- Are terms obtained from previous terms by multiplying by a particular amount?
- Are terms obtained by combining previous terms in a certain way?

The following examples illustrate how this type of problem might be attacked.

EXAMPLE 5 What is a rule that can produce the terms of a sequence if the first 10 terms are 1, 2, 2, 3, 3, 3, 4, 4, 4, 4?

Solution: Note that the integer 1 appears once, the integer 2 appears twice, the integer 3 appears three times, and the integer 4 appears four times. A reasonable rule for generating this sequence is that the integer n appears exactly n times, so the next five times of the sequence would all be 5, the following six terms would all be 6, and so on. The sequence generated this way is a possible match. ■

EXAMPLE 6 What is a rule that can produce the terms of a sequence if the first 10 terms are 5, 11, 17, 23, 29, 35, 41, 47, 53, 59?

Solution: Note that each of the first 10 terms of this sequence after the first is obtained by adding 6 to the previous term. (We could see this by noticing that the difference between consecutive terms is 6.) Consequently, the nth term could be produced by starting with 5 and adding 6 a total of $n - 1$ times; that is, a reasonable guess is that the nth term is $5 + 6(n - 1) = 6n - 1$ ■

The sequence in the solution of Example 6 is an **arithmetic progression,** which is a sequence of the form $a, a + d, a + 2d, a + 3d, \ldots, a + nd, \ldots$. In particular, this sequence has $a = 5$ and $d = 6$.

Another useful technique for finding a rule for generating the terms of a sequence is to compare the terms of a sequence of interest with the terms of a well-known integer sequence, such as terms of an arithmetic progression, terms of a geometric progression (see Example 12), perfect squares, perfect cubes, and so on. The first 10 terms of some sequences you may want to keep in mind are displayed in Table 1.

EXAMPLE 7 Conjecture a simple formula for a_n if the first 10 terms of the sequence $\{a_n\}$ are 1, 7, 25, 79, 341, 727, 2185, 6559, 19681, 59047.

Solution: To attack this problem, we begin by looking at the difference of consecutive terms, but we do not see a pattern. When we form the ratio of consecutive terms to see

TABLE 1 Some Useful Sequences.

nth Term	*First 10 Terms*
n^2	1, 4, 9, 16, 25, 36, 49, 64, 81, 100, ...
n^3	1, 8, 27, 64, 125, 216, 343, 512, 729, 1000, ...
n^4	1, 16, 81, 256, 625, 1296, 2401, 4096, 6561, 10000, ...
2^n	2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, ...
3^n	3, 9, 27, 81, 243, 729, 2187, 6561, 19683, 59049, ...
$n!$	1, 2, 6, 24, 120, 720, 5040, 40320, 362880, 3628800, ...

whether each term is a multiple of the previous term, we find that this ratio, although not a constant, is close to 3. So it is reasonable to suspect that the terms of this sequence are generated by a formula involving 3^n. Comparing these terms with the corresponding terms of the sequence $\{3^n\}$, we notice that the nth term is 2 less than the corresponding power of 3. We see that $a_n = 3^n - 2$ for $1 \le n \le 10$ and conjecture that this formula holds for all n. ■

We will see throughout this text that integer sequences appear in a wide range of contexts in discrete mathematics. Sequences we will encounter include the sequence of prime numbers (Chapter 2), the number of ways to order n discrete objects (Chapter 4), the number of the moves required to solve the famous Tower of Hanoi puzzle with n disks (Chapter 5), the number of rabbits on an island after n months (Chapter 5), and the number of comparisons needed to sort n numbers (Chapter 8).

Integer sequences appear in a fabulously wide range of subject areas besides discrete mathematics, including biology, physics, engineering, chemistry, and physics, as well as in puzzles. A wonderfully diverse collection of over 8000 different integer sequences has been constructed over the past 20 years by the mathematician Neil Sloane, who has teamed up with Simon Plouffe, to produce *The Encyclopedia of Integers Sequences* ([SlPl95]). An extended list of the sequences is available on the Web, with new sequences added regularly. There is also a program accessible via the Web that you can use to find sequences from the encyclopedia that match initial terms you provide.

Neil Sloane (born 1939). Neil Sloane studied mathematics and electrical engineering at the University of Melbourne on a scholarship from the Australian state telephone company. He mastered many telephone-related jobs, such as erecting telephone poles, in his summer work. After graduating, he designed minimal cost telephone networks in Australia. In 1962 he came to the United States and studied electrical engineering at Cornell University. His Ph.D. thesis was on what are now called neural networks. He took a job at Bell Labs in 1969, working in many areas, including network design, coding theory, and sphere packing. He now works for AT&T Labs, moving there from Bell Labs when AT&T split up in 1996. One of his favorite problems is the **kissing problem** (a name he coined), which asks how many spheres can be arranged in n dimensions so that they all touch a central sphere of the same size. (In two dimensions the answer is 6, since 6 pennies can be placed so they touch a central penny. In three dimensions, 12 billiard balls can be placed so that they touch a central billiard ball. Two billiard balls that just touch are said to "kiss," giving rise to the terminology "kissing problem" and "kissing number.") Sloane, together with Andrew Odlyzko, showed that in 8 and 24 dimensions the optimal kissing numbers are, respectively, 240 and 196,560. The kissing number is known in dimensions 1, 2, 3, 8, and 24, but not in any other dimensions. Sloane's books include *Sphere Packings, Lattices and Groups*, 3d ed., with John Conway; *The Theory of Error-Correcting Codes* with Jessie MacWilliams; *The Encyclopedia of Integer Sequences* with Simon Plouffe; and *The Rock-Climbing Guide to New Jersey Crags* with Paul Nick. The last book demonstrates his interest in rock climbing; it includes more than 50 climbing sites in New Jersey.

SUMMATIONS

Next, we introduce **summation notation.** We begin by describing the notation used to express the sum of the terms

$$a_m, a_{m+1}, \dots, a_n$$

from the sequence $\{a_n\}$. We use the notation

$$\sum_{j=m}^{n} a_j$$

to represent

$$a_m + a_{m+1} + \cdots + a_n.$$

Here, the variable j is called the **index of summation,** and the choice of the letter j as the variable is arbitrary; that is, we could have used any other letter, such as i or k. Or, in notation,

$$\sum_{j=m}^{n} a_j = \sum_{i=m}^{n} a_i = \sum_{k=m}^{n} a_k.$$

Here, the index of summation runs through all integers starting with its **lower limit** m and ending with its **upper limit** n. The uppercase Greek letter sigma, Σ, is used to denote summation. We give some examples of summation notation.

EXAMPLE 8 Express the sum of the first 100 terms of the sequence $\{a_n\}$, where $a_n = 1/n$ for $n = 1, 2, 3, \ldots$.

Solution: The lower limit for the index of summation is 1, and the upper limit is 100. We write this sum as

$$\sum_{j=1}^{100} \frac{1}{j}.$$

■

EXAMPLE 9 What is the value of $\sum_{j=1}^{5} j^2$?

Solution: We have

$$\begin{aligned}\sum_{j=1}^{5} j^2 &= 1^2 + 2^2 + 3^2 + 4^2 + 5^2 \\ &= 1 + 4 + 9 + 16 + 25 \\ &= 55.\end{aligned}$$

■

EXAMPLE 10 What is the value of $\sum_{k=4}^{8} (-1)^k$?

Solution: We have

$$\begin{aligned}\sum_{k=4}^{8} (-1)^k &= (-1)^4 + (-1)^5 + (-1)^6 + (-1)^7 + (-1)^8 \\ &= 1 + (-1) + 1 + (-1) + 1 \\ &= 1.\end{aligned}$$

■

Sometimes it is useful to shift the index of summation in a sum. This is often done when two sums need to be added but their indices of summation do not match. When shifting an index of summation, it is important to make the appropriate changes in the corresponding summand. This is illustrated by the following example.

EXAMPLE 11 Suppose we have the sum

$$\sum_{j=1}^{5} j^2$$

but want the index of summation to run between 0 and 4 rather than from 1 to 5. To do this, we let $k = j - 1$. Then the new summation index runs from 0 to 4, and the term j^2 becomes $(k + 1)^2$. Hence

$$\sum_{j=1}^{5} j^2 = \sum_{k=0}^{4} (k + 1)^2.$$

It is easily checked that both sums are $1 + 4 + 9 + 16 + 25 = 55$. ■

EXAMPLE 12 A *geometric progression* is a sequence of the form

$$a, ar, ar^2, ar^3, \ldots, ar^k,$$

where a, the initial term, and r, the common ratio, are real numbers. Sums of terms of geometric progressions commonly arise; such sums are called *geometric series*. We will find a formula for S, the sum of the first $n + 1$ terms of a geometric progression with initial term a and common nonzero ratio r; that is,

$$S = \sum_{j=0}^{n} ar^j.$$

To compute S, first multiply both sides of the equality by r and then manipulate the resulting sum as follows:

$$\begin{aligned} rS &= r\sum_{j=0}^{n} ar^j \\ &= \sum_{j=0}^{n} ar^{j+1} \\ &= \sum_{k=1}^{n+1} ar^k \\ &= \sum_{k=0}^{n} ar^k + (ar^{n+1} - a) && \text{(This equality is obtained by shifting the index of summation, setting } k = j + 1.) \\ &= S + (ar^{n+1} - a). \end{aligned}$$

From these equalities, we see that

$$rS = S + (ar^{n+1} - a).$$

Solving for S shows that if $r \neq 1$

$$S = \frac{ar^{n+1} - a}{r - 1}.$$

If $r = 1$, then clearly the sum equals $(n + 1)a$. ∎

EXAMPLE 13 Double summations arise in many contexts (as in the analysis of nested loops in computer programs). An example of a double summation is

$$\sum_{i=1}^{4}\sum_{j=1}^{3} ij.$$

To evaluate the double sum, first expand the inner summation and then continue by computing the outer summation:

$$\begin{aligned}\sum_{i=1}^{4}\sum_{j=1}^{3} ij &= \sum_{i=1}^{4}(i + 2i + 3i)\\ &= \sum_{i=1}^{4} 6i\\ &= 6 + 12 + 18 + 24 = 60.\end{aligned}$$ ∎

We can also use summation notation to add all values of a function, or terms of an indexed set, where the index of summation runs over all values in a set. That is, we write

$$\sum_{s \in S} f(s)$$

to represent the sum of the values $f(s)$, for all members s of S.

EXAMPLE 14 What is the value of $\sum_{s\in\{0,2,4\}} s$?

Solution: Since $\sum_{s\in\{0,2,4\}} s$ represents the sum of the values of s for all the members of the set $\{0, 2, 4\}$, it follows that

$$\sum_{s\in\{0,2,4\}} s = 0 + 2 + 4 = 6.$$ ∎

Certain sums arise repeatedly throughout discrete mathematics. Having a collection of formulae for such sums can be useful, so Table 2 provides a small table of formulae for commonly occurring sums.

We derived the first formula in this table in Example 12. The remaining three formulae give us the sum of the first n positive intergers, the sum of their squares, and the sum of their cubes. These three formulae can be derived in many different ways (for example, see Exercises 21 and 22 at the end of this section). Also note that each formula, once known, can easily be proved using mathematical induction, the subject of Section 3.2.

Example 15 illustrates how the formulae in Table 2 can be useful.

TABLE 2 Some Useful Summation Formulae.

Sum	*Closed Form*
$\sum_{k=0}^{n} ar^k$	$\frac{ar^{n+1}-a}{r-1}, r \neq 1$
$\sum_{k=1}^{n} k$	$\frac{n(n+1)}{2}$
$\sum_{k=1}^{n} k^2$	$\frac{n(n+1)(2n+1)}{6}$
$\sum_{k=1}^{n} k^3$	$\frac{n^2(n+1)^2}{4}$

EXAMPLE 15 Find $\sum_{k=50}^{100} k^2$.

Solution: First note that since $\sum_{k=1}^{100} k^2 = \sum_{k=1}^{49} k^2 + \sum_{k=50}^{100} k^2$, we have

$$\sum_{k=50}^{100} k^2 = \sum_{k=1}^{100} k^2 - \sum_{k=1}^{49} k^2.$$

Using the formula $\sum_{k=1}^{n} k^2 = n(n+1)(2n+1)/6$ from Table 2, we see that

$$\sum_{k=50}^{100} k^2 = \frac{100 \cdot 101 \cdot 201}{6} - \frac{49 \cdot 50 \cdot 99}{6} = 338{,}350 - 40{,}425 = 297{,}925.$$

■

CARDINALITY (optional)

Recall that in Section 1.4, the cardinality of a finite set was defined to be the number of elements in the set. It is possible to extend the concept of cardinality to all sets, both finite and infinite, with the following definition.

DEFINITION 2. The sets A and B have the same *cardinality* if and only if there is a one-to-one correspondence from A to B.

To see that this definition agrees with the previous definition of the cardinality of a finite set as the number of elements in that set, note that there is a one-to-one correspondence between any two finite sets with n elements, where n is a nonnegative integer.

We will now split infinite sets into two groups, those with the same cardinality as the set of natural numbers and those with different cardinality.

DEFINITION 3. A set that is either finite or has the same cardinality as the set of natural numbers is called *countable*. A set that is not countable is called *uncountable*.

We now give examples of countable and uncountable sets.

EXAMPLE 16 Show that the set of odd positive integers is a countable set.

Solution: To show that the set of odd positive integers is countable, we will exhibit a one-to-one correspondence between this set and the set of natural numbers. Consider

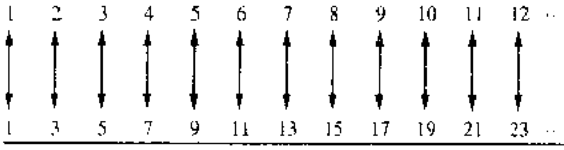


FIGURE 1 **A One-to-One Correspondence Between N and the Set of Odd Positive Integers.**

the function

$$f(n) = 2n - 1$$

from $\mathbf{N}$ to the set of odd positive integers. We show that f is a one-to-one correspondence by showing that it is both one-to-one and onto. To see that it is one-to-one, suppose that $f(n) = f(m)$. Then $2n - 1 = 2m - 1$, so that $n = m$. To see that it is onto, suppose that t is an odd positive integer. Then t is 1 less than an even integer $2k$, where k is a natural number. Hence $t = 2k - 1 = f(k)$. We display this one-to-one correspondence in Figure 1. ■

An infinite set is countable if and only if it is possible to list the elements of the set in a sequence (indexed by the natural numbers). The reason for this is that a one-to-one correspondence f from the set of natural numbers to a set S can be expressed in terms of a sequence $a_1, a_2, \ldots, a_n, \ldots$ where $a_1 = f(1), a_2 = f(2), \ldots, a_n = f(n), \ldots$. For instance, the set of odd integers can be listed in a sequence $a_1, a_2, \ldots, a_n, \ldots$, where $a_n = 2n - 1$.

We now give an example of an uncountable set.

EXAMPLE 17 Show that the set of real numbers is an uncountable set.

Solution: To show that the set of real numbers is uncountable, we suppose that the set of real numbers is countable and arrive at a contradiction. Then, the subset of all real numbers that fall between 0 and 1 would also be countable (since any subset of a countable set is also countable; see Exercise 32 at the end of the section). Under this assumption, the real numbers between 0 and 1 can be listed in some order, say, $r_1, r_2, r_3, \ldots$. Let the decimal representation of these real numbers be

$$\begin{aligned}
r_1 &= 0.d_{11}d_{12}d_{13}d_{14}\ldots \\
r_2 &= 0.d_{21}d_{22}d_{23}d_{24}\ldots \\
r_3 &= 0.d_{31}d_{32}d_{33}d_{34}\ldots \\
r_4 &= 0.d_{41}d_{42}d_{43}d_{44}\ldots \\
&\ \ \vdots
\end{aligned}$$

where $d_{ij} \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. (For example, if $r_1 = 0.23794102\ldots$, we have $d_{11} = 2, d_{12} = 3, d_{13} = 7$, and so on.) Then, form a new real number with decimal expansion $r = 0.d_1d_2d_3d_4\ldots$, where the decimal digits are determined by the following rule:

$$d_i = \begin{cases} 4 & \text{if } d_{ii} = 4 \\ 5 & \text{if } d_{ii} \neq 4. \end{cases}$$

(As an example, suppose that $r_1 = 0.23794102\ldots$, $r_2 = 0.44590138\ldots$, $r_3 = 0.09118764\ldots$, $r_4 = 0.80553900\ldots$, and so on. Then we have $r = 0.d_1d_2d_3d_4\ldots = 0.4544\ldots$, where $d_1 = 4$ since $d_{11} \neq 4$, $d_2 = 5$ since $d_{22} = 4$, $d_3 = 4$ since $d_{33} \neq 4$, $d_4 = 4$ since $d_{44} = 4$, and so on.)

Every real number has a unique decimal expansion (when the possibility that the expansion has a tail end that consists entirely of the digit 9 is excluded). Then, the real number r is not equal to any of $r_1, r_2, \ldots$, since the decimal expansion of r differs from the decimal expansion of r_i in the ith place to the right of the decimal point, for each i.

Since there is a real number r between 0 and 1 that is not in the list, the assumption that all the real numbers between 0 and 1 could be listed must be false. Therefore, all the real numbers between 0 and 1 cannot be listed, so that the set of real numbers between 0 and 1 is uncountable. Any set with an uncountable subset is uncountable (see Exercise 35 at the end of this section). Hence, the set of real numbers is uncountable. ■

Exercises

1. Find the following terms of the sequence $\{a_n\}$ where $a_n = 2 \cdot (-3)^n + 5^n$.
a) a_0 **b)** a_1 **c)** a_4 **d)** a_5

2. What is the term a_8 of the sequence $\{a_n\}$ if a_n equals
a) 2^{n-1}? **b)** 7?
c) $1 + (-1)^n$? **d)** $-(-2)^n$?

3. What are the terms a_0, a_1, a_2, and a_3 of the sequence $\{a_n\}$, where a_n equals
a) $2^n + 1$? **b)** $(n + 1)^{n+1}$?
c) $\lceil n/2 \rceil$? **d)** $\lfloor n/2 \rfloor + \lceil n/2 \rceil$?

4. What are the terms a_0, a_1, a_2, and a_3 of the sequence $\{a_n\}$, where a_n equals
a) $(-2)^n$? **b)** 3?
c) $7 + 4^n$? **d)** $2^n + (-2)^n$?

5. List the first 10 terms of each of the following sequences.
- **a)** the sequence that begins with 2 and in which each successive term is 3 more than the preceding term
- **b)** the sequence that lists each positive integer three times, in increasing order
- **c)** the sequence that lists the odd positive integers in increasing order, listing each odd integer twice
- **d)** the sequence whose nth term is $n! - 2^n$
- **e)** the sequence that begins with 3, where each succeeding term is twice the preceding term
- **f)** the sequence whose first two terms are 1 and each succeeding term is the sum of the two preceding terms (This is the famous Fibonacci sequence, which we will study later in this text.)
- **g)** the sequence whose nth term is the number of bits in the binary expansion of the number n (defined in Section 2.3)
- **h)** the sequence where the nth term is the number of letters in the English word for the index n

6. List the first 10 terms of each of the following sequences.
- **a)** the sequence obtained by starting with 10 and obtaining each term by subtracting 3 from the previous term
- **b)** the sequence whose nth term is the sum of the first n positive integers
- **c)** the sequence whose nth term is $3^n - 2^n$
- **d)** the sequence whose nth term is $\lfloor \sqrt{n} \rfloor$
- **e)** the sequence whose first two terms are 1 and 2 and each succeeding term is the sum of the two previous terms
- **f)** the sequence whose nth term is the largest integer whose binary expansion (defined in Section 2.3) has n bits. (Write your answer in decimal notation.)
- **g)** the sequence whose terms are constructed sequentially as follows: start with 1, then add 1, then multiply by 1, then add 2, then multiply by 2, and so on
- **h)** the sequence whose nth term is the largest integer k such that $k! \leq n$

7. Find at least three different sequences beginning with the terms 1, 2, 4 whose terms are generated by a simple formula or rule.

8. Find at least three different sequences beginning with the terms 3, 5, 7 whose terms are generated by a simple formula or rule.

9. For each of the following lists of integers, provide a simple formula or rule that generates the terms of an integer sequence that begins with the given list.
- **a)** 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, . . .
- **b)** 1, 2, 2, 3, 4, 4, 5, 6, 6, 7, 8, 8, . . .
- **c)** 1, 0, 2, 0, 4, 0, 8, 0, 16, 0, . . .
- **d)** 3, 6, 12, 24, 48, 96, 192, . . .

e) 15, 8, 1, −6, −13, −20, −27, ...
f) 3, 5, 8, 12, 17, 23, 30, 38, 47, ...
g) 2, 16, 54, 128, 250, 432, 686, ...
h) 2, 3, 7, 25, 121, 721, 5041, 40321, ...

10. For each of the following lists of integers, provide a simple formula or rule that generates the terms of an integer sequence that begins with the given list.
a) 3, 6, 11, 18, 27, 38, 51, 66, 83, 102, ...
b) 7, 11, 15, 19, 23, 27, 31, 35, 39, 43, ...
c) 1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010, 1011, ...
d) 1, 2, 2, 2, 3, 3, 3, 3, 3, 5, 5, 5, 5, 5, 5, 5, ...
e) 0, 2, 8, 26, 80, 242, 728, 2186, 6560, 19682, ...
f) 1, 3, 15, 105, 945, 10395, 135135, 2027025, 34459425, ...
g) 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, ...
h) 2, 4, 16, 256, 65536, 4294967296, ...

***11.** Show that if a_n denotes the nth positive integer that is not a perfect square, then $a_n = n + \{\sqrt{n}\}$, where $\{x\}$ denotes the integer closest to the real number x.

***12.** Let a_n be the nth term of the sequence 1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, ..., constructed by including the integer k exactly k times. Show that $a_n = \lfloor \sqrt{2n} + \frac{1}{2} \rfloor$.

13. What are the values of the following sums?
a) $\sum_{k=1}^{5}(k+1)$ **b)** $\sum_{j=0}^{4}(-2)^j$
c) $\sum_{i=1}^{10} 3$ **d)** $\sum_{j=0}^{8}(2^{j+1}-2^j)$

14. What are the values of the following sums, where $S = \{1, 3, 5, 7\}$?
a) $\sum_{j \in S} j$ **b)** $\sum_{j \in S} j^2$
c) $\sum_{j \in S}(1/j)$ **d)** $\sum_{j \in S} 1$

15. What is the value of each of the following sums of terms of a geometric progression?
a) $\sum_{j=0}^{8} 3 \cdot 2^j$ **b)** $\sum_{j=1}^{8} 2^j$
c) $\sum_{j=2}^{8}(-3)^j$ **d)** $\sum_{j=0}^{8} 2 \cdot (-3)^j$

16. Find the value of each of the following sums.
a) $\sum_{j=0}^{8}(1+(-1)^j)$ **b)** $\sum_{j=0}^{8}(3^j-2^j)$
c) $\sum_{j=0}^{8}(2 \cdot 3^j + 3 \cdot 2^j)$ **d)** $\sum_{j=0}^{8}(2^{j+1}-2^j)$

17. Compute each of the following double sums.
a) $\sum_{i=1}^{2}\sum_{j=1}^{3}(i+j)$ **b)** $\sum_{i=0}^{2}\sum_{j=0}^{3}(2i+3j)$
c) $\sum_{i=1}^{3}\sum_{j=0}^{2} i$ **d)** $\sum_{i=0}^{2}\sum_{j=1}^{3} ij$

18. Compute each of the following double sums.
a) $\sum_{i=1}^{3}\sum_{j=1}^{2}(i-j)$ **b)** $\sum_{i=0}^{3}\sum_{j=0}^{2}(3i+2j)$
c) $\sum_{i=1}^{3}\sum_{j=0}^{2} j$ **d)** $\sum_{i=0}^{2}\sum_{j=0}^{3} i^2 j^3$

19. Show that $\sum_{j=1}^{n}(a_j - a_{j-1}) = a_n - a_0$ where $a_0, a_1, \ldots, a_n$ is a sequence of real numbers. This type of sum is called **telescoping.**

20. Use the identity $1/(k(k+1)) = 1/k - 1/(k+1)$ and Exercise 19 to compute $\sum_{k=1}^{n} 1/(k(k+1))$.

21. Sum both sides of the identity $k^2 - (k-1)^2 = 2k - 1$ from $k = 1$ to $k = n$ and use Exercise 19 to find
a) a formula for $\sum_{k=1}^{n}(2k-1)$ (the sum of the first n odd natural numbers).
b) a formula for $\sum_{k=1}^{n} k$.

***22.** Use the technique given in Exercise 19, together with the result of Exercise 13b, to find a formula for $\sum_{k=1}^{n} k^2$.

23. Find $\sum_{k=100}^{200} k$. (Use Table 2.)

24. Find $\sum_{k=99}^{200} k^3$. (Use Table 2.)

***25.** Find a formula for $\sum_{k=0}^{m}\lfloor\sqrt{k}\rfloor$, when m is a positive integer. (*Hint:* Use the formula for $\sum_{k=1}^{n} k^2$.)

***26.** Find a formula for $\sum_{k=0}^{m}\lfloor\sqrt[3]{k}\rfloor$, when m is a positive integer. (*Hint:* Use the formula for $\sum_{k=1}^{n} k^3$.)

There is also a special notation for products. The product of $a_m, a_{m+1}, \ldots, a_n$ is represented by

$$\prod_{j=m}^{n} a_j.$$

27. What are the values of the following products?
a) $\prod_{i=0}^{10} i$ **b)** $\prod_{i=5}^{8} i$
c) $\prod_{i=1}^{100}(-1)^i$ **d)** $\prod_{i=1}^{10} 2$

The value of the **factorial function** at a positive integer n, denoted by $n!$, is the product of the positive integers from 1 to n, inclusive. Also, we specify that $0! = 1$.

28. Express $n!$ using product notation.

29. Find $\sum_{j=0}^{4} j!$.

30. Find $\prod_{j=0}^{4} j!$.

31. Determine whether each of the following sets is countable or uncountable. For those that are countable, exhibit a one-to-one correspondence between the set of natural numbers and that set.
a) the negative integers
b) the even integers
c) the real numbers between 0 and $\frac{1}{2}$
d) integers that are multiples of 7

***32.** Determine whether each of the following sets is countable or uncountable. For those that are countable, exhibit a one-to-one correspondence between the set of natural numbers and that set.
a) integers not divisible by 3
b) integers divisible by 5 but not by 7

c) the real numbers with decimal representations consisting of all 1s

d) the real numbers with decimal representations of all 1s or 9s

33. If A is an uncountable set and B is a countable set, must $A - B$ be uncountable?

34. Show that a subset of a countable set is also countable.

35. Show that if A is an uncountable set and $A \subseteq B$, then B is uncountable.

36. Show that the union of two countable sets is countable.

****37.** Show that the union of a countable number of countable sets is countable.

***38.** A real number is called **rational** if it can be written as the quotient of two integers. Show that the set of rational numbers between 0 and 1 is countable. (*Hint:* List the elements of this set in order of increasing $p + q$, where p is the numerator and q is the denominator of a fraction p/q in lowest terms.)

***39.** Show that the set of all bit strings is countable.

***40.** Show that the set of real numbers that are solutions of quadratic equations $ax^2 + bx + c = 0$, where a, b, and c are integers, is countable.

***41.** Show that the set of all computer programs in a particular programming language is countable. (*Hint:* A computer program written in a programming language can be thought of as a string of symbols from a finite alphabet.)

***42.** Show that the set of functions from the positive integers to the set $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ is uncountable. (*Hint:* First set up a one-to-one correspondence between the set of real numbers between 0 and 1 and a subset of these functions. Do this by associating to the real number $0.d_1d_2\ldots d_n\ldots$ the function f with $f(n) = d_n$.)

***43.** We say that a function is **computable** if there is a computer program that finds the values of this function. Use Exercises 41 and 42 to show that there are functions that are not computable.

1.8

The Growth of Functions

INTRODUCTION

Suppose that a computer program reorders any list of n integers into a list where the integers are in increasing order. One important consideration concerning the practicality of this program is how long a computer takes to solve this problem. An analysis may show that the time used to reorder a list of n integers (where these integers are less than some specified size) is less than $f(n)$ microseconds, where $f(n) = 100n \log n + 25n + 9$. To analyze the practicality of the program, we need to understand how quickly this function grows as n grows. This section reviews some important methods used in estimating the growth of functions. We will introduce the notation most commonly used in the analysis of the growth of functions, namely, **big-*O*** notation. We will develop some useful results about the growth of functions using this notation.

BIG-*O* NOTATION

The growth of functions is often described using a special notation. The following definition describes this notation.

DEFINITION 1. Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x)$ is $O(g(x))$ if there are constants C and k such that

$$|f(x)| \le C|g(x)|$$

whenever $x > k$. (This is read as "$f(x)$ is big-oh of $g(x)$.")

Remark: To show $f(x)$ is $O(g(x))$, we need only find *one* pair of constants C and k such that $|f(x)| \leq C|g(x)|$ if $x > k$. However, a pair C, k that satisfies the definition is *never* unique. Moreover, if one such pair exists, there are *infinitely many* such pairs. A simple way to see this is to note that if C, k is one such pair, any pair C', k' with $C < C'$ and $k < k'$ also satisfies the definition, since $|f(x)| \leq C|g(x)| \leq C'|g(x)|$ whenever $x > k' > k$.

EXAMPLE 1 Show that $f(x) = x^2 + 2x + 1$ is $O(x^2)$.

Solution: Since

$$0 \leq x^2 + 2x + 1 \leq x^2 + 2x^2 + x^2 = 4x^2$$

whenever $x > 1$, it follows that $f(x)$ is $O(x^2)$. (To apply the definition of big-O notation here, take $C = 4$ and $k = 1$. It is not necessary to use absolute values here since all functions in these equalities are positive when x is positive.)

Another approach is to note that when $x > 2$, it follows that $2x \leq x^2$. Consequently, if $x > 2$, we see that

$$0 \leq x^2 + 2x + 1 \leq x^2 + x^2 + x^2 = 3x^2.$$

(We apply the definition with $C = 3$ and $k = 2$.)

Observe that in the relationship $f(x)$ is $O(x^2)$, x^2 can be replaced by any function with larger values than x^2, for example, $f(x)$ is $O(x^3)$, $f(x)$ is $O(x^2 + 2x + 7)$, and so on. It is also true that x^2 is $O(x^2 + 2x + 1)$, since $x^2 < x^2 + 2x + 1$ whenever $x \geq 1$.

Figure 1 illustrates that $x^2 + 2x + 1$ is $O(x^2)$. ∎

Note that in Example 1 we have two functions, $f(x) = x^2 + 2x + 1$ and $g(x) = x^2$, such that $f(x)$ is $O(g(x))$ and $g(x)$ is $O(f(x))$—the latter fact following from the inequality $x^2 \leq x^2 + 2x + 1$, which holds for all nonnegative real numbers x. We say that two functions $f(x)$ and $g(x)$ that satisfy both of these big-O relationships are of the *same order.* (See pages 88–90.)

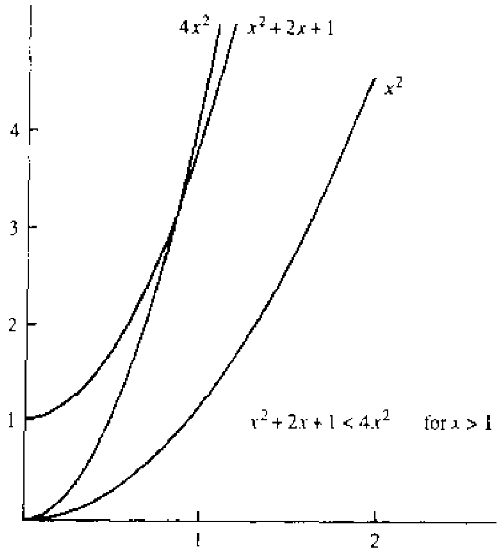


FIGURE 1 The Function $x^2 + 2x + 1$ Is $O(x^2)$.

Remark: The fact that $f(x)$ is $O(g(x))$ is sometimes written $f(x) = O(g(x))$. However, the equals sign in this notation does *not* represent a genuine equality. Rather, this notation tells us that an inequality holds relating the values of the functions f and g for sufficiently large numbers in the domains of these functions.

Big-O notation has been used in mathematics for almost a century. In computer science it is widely used in the analysis of algorithms, as will be seen in Chapter 2. The German mathematician Paul Bachmann first introduced big-O notation in 1892 in an important book on number theory. The big-O symbol is sometimes called a **Landau symbol** after the German mathematician Edmund Landau, who used this notation throughout his work. The use of big-O notation in computer science was popularized by Donald Knuth, who also introduced the big-Omega and big-Theta notations defined later in this section.

When $f(x)$ is $O(g(x))$, and $h(x)$ is a function that has larger absolute values than $g(x)$ does for sufficiently large values of x, it follows that $f(x)$ is $O(h(x))$. In other words, the function $g(x)$ in the relationship $f(x)$ is $O(g(x))$ can be replaced by a function with larger absolute values. To see this, note that if

$$|f(x)| \leq C|g(x)| \qquad \text{if } x > k,$$

and if $|h(x)| > |g(x)|$ for all $x > k$, then

$$|f(x)| \leq C|h(x)| \qquad \text{if } x > k.$$

Hence, $f(x)$ is $O(h(x))$.

When big-O notation is used, the function g in the relationship $f(x)$ is $O(g(x))$ is chosen to be as small as possible (sometimes from a set of reference functions, such as functions of the form x^n, where n is a positive integer).

In subsequent discussions, we will almost always deal with functions that take on only positive values. All references to absolute values can be dropped when working with big-O estimates for such functions. Figure 2 illustrates the relationship $f(x)$ is $O(g(x))$.

The following example illustrates how big-O notation is used to estimate the growth of functions.

Paul Gustav Heinrich Bachmann (1837–1920). Paul Bachmann, the son of a Lutheran pastor, shared his father's pious lifestyle and love of music. His mathematical talent was discovered by one of his teachers, even though he had difficulties with some of his early mathematical studies. After recuperating from tuberculosis in Switzerland, Bachmann studied mathematics, first at the University of Berlin and later at Göttingen, where he attended lectures presented by the famous number theorist Dirichlet. He received his doctorate under the German number theorist Kummer in 1862; his thesis was on group theory. Bachmann was a professor at Breslau and later at Münster. After he retired from his professorship, he continued his mathematical writing, played the piano, and served as a music critic for newspapers. Bachmann's mathematical writings include a five-volume survey of results and methods in number theory, a two-volume work on elementary number theory, a book on irrational numbers, and a book on the famous conjecture known as Fermat's Last Theorem. He introduced big-O notation in his 1892 book *Analytische Zahlentheorie.*

Edmund Landau (1877–1938). Edmund Landau, the son of a Berlin gynecologist, attended high school and university in Berlin. He received his doctorate in 1899, under the direction of Frobenius. Landau first taught at the University of Berlin and then moved to Göttingen, where he was a full professor until the Nazis forced him to stop teaching. Landau's main contributions to mathematics were in the field of analytic number theory. In particular, he established several important results concerning the distribution of primes. He authored a three-volume exposition on number theory as well as other books on number theory and mathematical analysis.

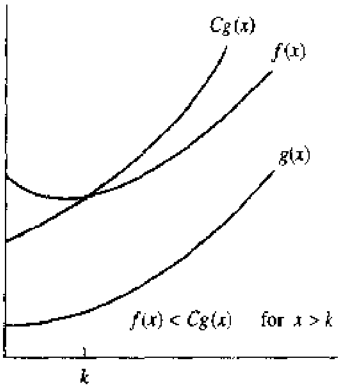


FIGURE 2 **The Function $f(x)$ Is $O(g(x))$.**

Donald E. Knuth (born 1938). Knuth grew up in Milwaukee, where his father taught bookkeeping at a Lutheran high school and owned a small printing business. He was an excellent student, setting academic achievement awards. He applied his intelligence in unconventional ways, winning a contest when he was in the eighth grade by finding over 4,500 words that could be formed from the letters in "Ziegler's Giant Bar." This won a television set for his school and a candy bar for everyone in his class.

Knuth had a difficult time choosing physics over music as his major at the Case Institute of Technology. He then switched from physics to mathematics, and in 1960 he received his bachelor of science degree, simultaneously receiving a master of science degree by a special award of the faculty who considered his work outstanding. At Case, he managed the basketball team and applied his talents by constructing a formula for the value of each player. This novel approach was covered by *Newsweek* and by Walter Cronkite on the CBS television network. Knuth began graduate work at the California Institute of Technology in 1960 and received his Ph.D. there in 1963. During this time he worked as a consultant, writing compilers for different computers.

Knuth joined the staff of the California Institute of Technology in 1963, where he remained until 1968, when he took a job as a full professor at Stanford University. He retired as Professor Emeritus in 1992 to concentrate on writing. He is especially interested in updating and completing new volumes of his series *The Art of Computer Programming*, a work that has had a profound influence on the development of computer science, which he began writing as a graduate student in 1962, focusing on compilers. In common jargon, "Knuth," referring to *The Art of Computer Programming*, has come to mean the reference that answers all questions about such topics as data structures and algorithms.

Knuth is the founder of the modern study of computational complexity. He has made fundamental contributions to the subject of compilers. His dissatisfaction with mathematics typography sparked him to invent the now widely used TeX and Metafont systems. TeX has become a standard language for computer typography. Two of the many awards Knuth has received are the 1974 Turing Award and the 1979 National Medal of Technology, awarded to him by President Carter.

Knuth has written for a wide range of professional journals in computer science and in mathematics. However, his first publication, in 1957, when he was a college freshman, was a parody of the metric system called "The Potrzebie Systems of Weights and Measures," which appeared in *MAD Magazine* and has been in reprint several times. He is a church organist, as his father was. He is also a composer of music for the organ. Knuth believes that writing computer programs can be an aesthetic experience, much like writing poetry or composing music.

Knuth pays \$2.56 for the first person to find each error in his books and \$0.32 for significant suggestions. If you send him a letter with an error (you will need to use regular mail, since he has given up reading e-mail), he will eventually inform you whether you were the first person to tell him about this error. Be prepared for a long wait, since he receives an overwhelming amount of mail. (The author received a letter years after sending an error report to Knuth, noting that this report arrived several months after the first report of this error.)

EXAMPLE 2 Show that $7x^2$ is $O(x^3)$.

Solution: The inequality $7x^2 < x^3$ holds whenever $x > 7$. (We see this by dividing both sides of this inequality by x^2.) Hence, $7x^2$ is $O(x^3)$, taking $C = 1$ and $k = 7$ in the definition of big-O notation. ■

EXAMPLE 3 Example 2 shows that $7x^2$ is $O(x^3)$. Is it also true that x^3 is $O(7x^2)$?

Solution: To determine whether x^3 is $O(7x^2)$, it is necessary to determine whether there are constants C and k such that $x^3 \leq C(7x^2)$ whenever $x > k$. This inequality is equivalent to the inequality $x < 7C$, which is obtained by dividing both sides by x^2. No such C can exist since x can be made arbitrarily large. Hence x^3 is *not* $O(7x^2)$. ■

Polynomials can often be used to estimate the growth of functions. Instead of analyzing the growth of polynomials each time they occur, we would like a result that can always be used to estimate the growth of a polynomial. The following theorem does this. It shows that the leading term of a polynomial dominates its growth by asserting that a polynomial of degree n or less is $O(x^n)$.

THEOREM 1 **Let $f(x) = a_nx^n + a_{n-1}x^{n-1} + \cdots + a_1x + a_0$, where $a_0, a_1, \ldots, a_{n-1}, a_n$ are real numbers. Then $f(x)$ is $O(x^n)$.**

Proof: Using the triangle inequality, if $x > 1$ we have

$$\begin{aligned} |f(x)| &= |a_nx^n + a_{n-1}x^{n-1} + \cdots + a_1x + a_0| \\ &\leq |a_n|x^n + |a_{n-1}|x^{n-1} + \cdots + |a_1|x + |a_0| \\ &= x^n\left(|a_n| + |a_{n-1}|/x + \cdots + |a_1|/x^{n-1} + |a_0|/x^n\right) \\ &\leq x^n(|a_n| + |a_{n-1}| + \cdots + |a_1| + |a_0|). \end{aligned}$$

This shows that

$$|f(x)| \leq Cx^n$$

where $C = |a_n| + |a_{n-1}| + \cdots + |a_0|$ whenever $x > 1$. Hence, $f(x)$ is $O(x^n)$. □

We now give some examples involving functions that have the set of positive integers as their domains.

EXAMPLE 4 How can big-O notation be used to estimate the sum of the first n positive integers?

Solution: Since each of the integers in the sum of the first n positive integers does not exceed n, it follows that

$$1 + 2 + \cdots + n \leq n + n + \cdots + n = n^2.$$

From this inequality it follows that $1 + 2 + 3 + \cdots + n$ is $O(n^2)$, taking $C = 1$ and $k = 1$ in the definition of big-O notation. (In this example the domains of the functions in the big-O relationship are the set of positive integers.) ■

In the next example big-O estimates will be developed for the factorial function and its logarithm. These estimates will be important in the analysis of the number of steps used in sorting procedures.

EXAMPLE 5 Give big-O estimates for the factorial function and the logarithm of the factorial function, where the factorial function $f(n) = n!$ is defined by

$$n! = 1 \cdot 2 \cdot 3 \cdot \cdots \cdot n$$

whenever n is a positive integer, and $0! = 1$. For example,

$$1! = 1, \quad 2! = 1 \cdot 2 = 2, \quad 3! = 1 \cdot 2 \cdot 3 = 6, \quad 4! = 1 \cdot 2 \cdot 3 \cdot 4 = 24.$$

Note that the function $n!$ grows rapidly. For instance,

$$20! = 2{,}432{,}902{,}008{,}176{,}640{,}000.$$

Solution: A big-O estimate for $n!$ can be obtained by noting that each term in the product does not exceed n. Hence,

$$\begin{aligned} n! &= 1 \cdot 2 \cdot 3 \cdot \cdots \cdot n \\ &\leq n \cdot n \cdot n \cdot \cdots \cdot n \\ &= n^n. \end{aligned}$$

This inequality shows that $n!$ is $O(n^n)$. Taking logarithms of both sides of the inequality established for $n!$, we obtain

$$\log n! \leq \log n^n = n \log n.$$

This implies that $\log n!$ is $O(n \log n)$. ■

EXAMPLE 6 In Section 3.2 we will show that

$$n < 2^n$$

whenever n is a positive integer. Using this inequality we can conclude that n is $O(2^n)$. (Take $k = C = 1$ in the definition of big-O notation.) Since the logarithm function is increasing, taking logarithms (base 2) of both sides of this inequality shows that

$$\log n < n.$$

It follows that

$$\log n \text{ is } O(n).$$

(Again we take $C = k = 1$ in the definition of big-O notation.)

If we have logarithms to a base b, where b is different from 2, we still have $\log_b n$ is $O(n)$ since

$$\log_b n = \frac{\log n}{\log b} < \frac{n}{\log b}$$

whenever n is a positive integer. (We have used Theorem 3 in Appendix 1 to see that $\log_b n = \log n / \log b$). ■

THE GROWTH OF COMBINATIONS OF FUNCTIONS

Many algorithms are made up of two or more separate subprocedures. The number of steps used by a computer to solve a problem with input of a specified size using such an algorithm is the sum of the number of steps used by these subprocedures. To give a big-O estimate for the number of steps needed, it is necessary to find big-O estimates for the number of steps used by each subprocedure and then combine these estimates.

Big-O estimates of combinations of functions can be provided if care is taken when different big-O estimates are combined. In particular, it is often necessary to estimate the growth of the sum and the product of two functions. What can be said if big-O estimates for each of two functions are known? To see what sort of estimates hold for the sum and the product of two functions, suppose that $f_1(x)$ is $O(g_1(x))$ and $f_2(x)$ is $O(g_2(x))$.

From the definition of big-O notation, there are constants C_1, C_2, k_1, and k_2 such that

$$|f_1(x)| \leq C_1|g_1(x)|$$

when $x > k_1$, and

$$|f_2(x)| \leq C_2|g_2(x)|$$

when $x > k_2$. To estimate the sum of $f_1(x)$ and $f_2(x)$, note that

$$\begin{aligned}|(f_1 + f_2)(x)| &= |f_1(x) + f_2(x)|\\ &\leq |f_1(x)| + |f_2(x)| \quad \text{(using triangle inequality } |a + b| \leq |a| + |b|\text{).}\end{aligned}$$

When x is greater than both k_1 and k_2, it follows from the inequalities for $|f_1(x)|$ and $|f_2(x)|$ that

$$\begin{aligned}|f_1(x)| + |f_2(x)| &< C_1|g_1(x)| + C_2|g_2(x)|\\ &\leq C_1|g(x)| + C_2|g(x)|\\ &= (C_1 + C_2)|g(x)|\\ &= C|g(x)|,\end{aligned}$$

where $C = C_1 + C_2$ and $g(x) = \max(|g_1(x)|, |g_2(x)|)$. (Here $\max(a, b)$ denotes the maximum, or larger, of a and b.)

This inequality shows that $|(f_1 + f_2)(x)| \leq C|g(x)|$ whenever $x > k$, where $k = \max(k_1, k_2)$. We state this useful result as the following theorem.

THEOREM 2 **Suppose that $f_1(x)$ is $O(g_1(x))$ and $f_2(x)$ is $O(g_2(x))$. Then $(f_1 + f_2)(x)$ is $O(\max(g_1(x), g_2(x)))$.**

We often have big-O estimates for f_1 and f_2 in terms of the same function g. In this situation, Theorem 2 can be used to show that $(f_1 + f_2)(x)$ is also $O(g(x))$, since $\max(g(x), g(x)) = g(x)$. This result is stated in the following corollary.

COROLLARY 1 **Suppose that $f_1(x)$ and $f_2(x)$ are both $O(g(x))$. Then $(f_1 + f_2)(x)$ is $O(g(x))$.**

In a similar way big-O estimates can be derived for the product of the functions f_1 and f_2. When x is greater than $\max(k_1, k_2)$ it follows that

$$\begin{aligned}|(f_1 f_2)(x)| &= |f_1(x)||f_2(x)| \\ &\le C_1|g_1(x)|C_2|g_2(x)| \\ &\le C_1 C_2|(g_1 g_2)(x)| \\ &\le C|(g_1 g_2)(x)|,\end{aligned}$$

where $C = C_1 C_2$. From this inequality, it follows that $f_1(x)f_2(x)$ is $O(g_1 g_2)$, since there are constants C and k—namely, $C = C_1 C_2$ and $k = \max(k_1, k_2)$, since $|(f_1 f_2)(x)| \le C|g_1(x)g_2(x)|$ whenever $x > k$. This result is stated in the following theorem.

THEOREM 3 **Suppose that $f_1(x)$ is $O(g_1(x))$ and $f_2(x)$ is $O(g_2(x))$. Then $(f_1 f_2)(x)$ is $O(g_1(x)g_2(x))$.**

The goal in using big-O notation to estimate functions is to choose a function $g(x)$ that grows relatively slowly so that $f(x)$ is $O(g(x))$. The following examples illustrate how to use Theorems 2 and 3 to do this. The type of analysis given in these examples is often used in the analysis of the time used to solve problems using computer programs.

EXAMPLE 7 Give a big-O estimate for $f(n) = 3n\log(n!) + (n^2 + 3)\log n$, where n is a positive integer.

Solution: First, the product $3n\log(n!)$ will be estimated. From Example 5 we know that $\log(n!)$ is $O(n\log n)$. Using this estimate and the fact that $3n$ is $O(n)$, Theorem 3 gives the estimate that $3n\log(n!)$ is $O(n^2\log n)$.

Next, the product $(n^2 + 3)\log n$ will be estimated. Since $(n^2 + 3) < 2n^2$ when $n > 2$, it follows that $n^2 + 3$ is $O(n^2)$. Thus, from Theorem 3 it follows that $(n^2 + 3)\log n$ is $O(n^2\log n)$. Using Theorem 2 to combine the two big-O estimates for the products shows that $f(n) = 3n\log(n!) + n^2\log n$ is $O(n^2\log n)$. ■

EXAMPLE 8 Give a big-O estimate for $f(x) = (x + 1)\log(x^2 + 1) + 3x^2$.

Solution: First, a big-O estimate for $(x + 1)\log(x^2 + 1)$ will be found. Note that $(x + 1)$ is $O(x)$. Furthermore, $x^2 + 1 \le 2x^2$ when $x > 1$. Hence,

$$\log(x^2 + 1) \le \log(2x^2) = \log 2 + \log x^2 = \log 2 + 2\log x \le 3\log x,$$

if $x > 2$. This shows that $\log(x^2 + 1)$ is $O(\log x)$.

From Theorem 3 it follows that $(x + 1)\log(x^2 + 1)$ is $O(x\log x)$. Since $3x^2$ is $O(x^2)$, Theorem 2 tells us that $f(x)$ is $O(\max(x\log x, x^2))$. Since $x\log x \le x^2$, for $x > 1$, it follows that $f(x)$ is $O(x^2)$. ■

As mentioned before, big-O notation is used to estimate the number of operations needed to solve a problem using a specified procedure or algorithm. The functions used in these estimates often include the following:

1. $\log n, \quad n, \quad n\log n, \quad n^2, \quad 2^n, \quad n!$

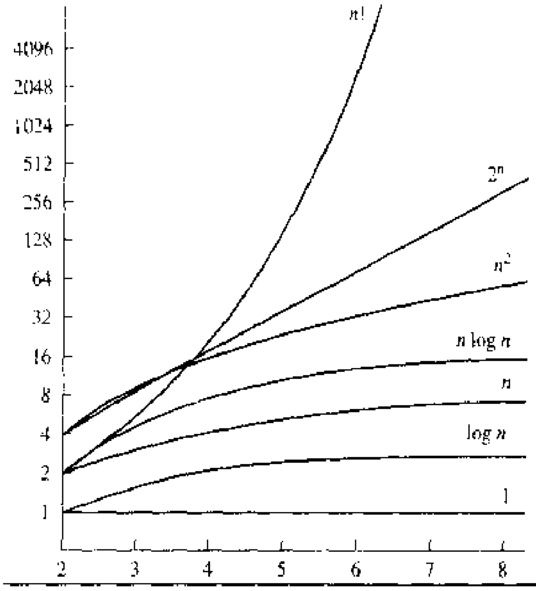


FIGURE 3 A Display of the Growth of Functions Commonly Used in Big-O Estimates.

Using calculus it can be shown that each function in the list is smaller than the succeeding function, in the sense that the ratio of a function and the succeeding function tends to zero as n grows without bound. Figure 3 displays the graphs of these functions, using a scale for the values of the functions that doubles for each successive marking on the graph.

BIG-OMEGA AND BIG-THETA NOTATION

Big-O notation is used extensively to describe the growth of functions, but it has limitations. In particular, when $f(x)$ is $O(g(x))$, we have an upper bound, in terms of $g(x)$, for the size of $f(x)$ for large values of x. However, big-O notation does not provide a lower bound for the size of $f(x)$ for large x. For this, we use **big-Omega notation.** When we want to give both an upper and a lower bound on the size of a function $f(x)$, relative to a reference function $g(x)$, we use **big-Theta notation.** Both big-Omega and big-Theta notation were introduced by Donald Knuth in the 1970s. His motivation for introducing these notations was the common misuse of big-O notation when both an upper and a lower bound on the size of a function are needed.

We now define big-Omega notation and illustrate its use. After doing so, we will do the same for big-Theta notation.

There is a strong connection between big-O and big-Omega notation. In particular, $f(x)$ is $\Omega(g(x))$ if and only if $g(x)$ is $O(f(x))$. We leave the verification of this fact as a straightforward exercise for the reader.

DEFINITION 2. Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x)$ is $\Omega(g(x))$ if there are positive constants C and k such that

$$|f(x)| \geq C|g(x)|$$

whenever $x > k$. (This is read as "$f(x)$ is big-Omega of $g(x)$".)

EXAMPLE 9 The function $f(x) = 8x^3 + 5x^2 + 7$ is $\Omega(g(x))$, where $g(x)$ is the function $g(x) = x^3$. This is easy to see since $f(x) = 8x^3 + 5x^2 + 7 \geq 8x^3$ for all positive real numbers x. This is equivalent to saying that $g(x) = x^3$ is $O(8x^3 + 5x^2 + 7)$, which can be established directly by turning the inequality around. ■

Often, it is important to know the order of growth of a function in terms of some relatively simple reference function such as x^n when n is a positive integer or c^x, where $c > 1$. Knowing the order of growth requires that we have both an upper bound and a lower bound for the size of the function. That is, given a function $f(x)$, we want a reference function $g(x)$ such that $f(x)$ is $O(g(x))$ and $f(x)$ is $\Omega(g(x))$. Big-Theta notation, defined as follows, is used to express both of these relationships, providing both an upper and a lower bound on the size of a function.

DEFINITION 3. Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x)$ is $\Theta(g(x))$ if $f(x)$ is $O(g(x))$ and $f(x)$ is $\Omega(g(x))$. When $f(x)$ is $\Theta(g(x))$, we say that "f is big-Theta of $g(x)$" and we also say that $f(x)$ is of order $g(x)$.

When $f(x)$ is $\Theta(g(x))$, it is also the case that $g(x)$ is $\Theta(f(x))$. Usually, when big-Theta notation is used, the function $g(x)$ in $\Theta(g(x))$ is a relatively simple reference function, such as x^n, c^x, $\log x$, and so on, while $f(x)$ can be relatively complicated.

EXAMPLE 10 We showed (in Example 4) that the sum of the first n positive integers is $O(n^2)$. Is this sum of order n^2?

Solution: Let $f(n) = 1 + 2 + 3 + \cdots + n$. Since we already know that $f(n) = O(n^2)$, to show that $f(n)$ is of order n^2 we need to find a positive constant C such that $f(n) > Cn^2$ for sufficiently large integers n. To obtain a lower bound for this sum, we can ignore the first half of the terms. Summing only the terms greater than $\lceil n/2 \rceil$, we find that

$$\begin{aligned}
1 + 2 + \cdots + n &\geq \lceil n/2 \rceil + (\lceil n/2 \rceil + 1) + \cdots + n \\
&\geq \lceil n/2 \rceil + \lceil n/2 \rceil + \cdots + \lceil n/2 \rceil \\
&= (n - \lceil n/2 \rceil + 1)\lceil n/2 \rceil \\
&\geq (n/2)(n/2) \\
&= n^2/4.
\end{aligned}$$

This shows that $f(n)$ is $\Omega(n^2)$. We conclude that $f(n)$ is of order n^2, or in symbols, $f(n)$ is $\Theta(n^2)$. ■

We can show that $f(x)$ is $\Theta(g(x))$ if we can find positive real numbers C_1 and C_2 and a positive real number k such that

$$C_1|g(x)| \leq |f(x)| \leq C_2|g(x)|$$

whenever $x \geq k$. This shows that $f(x)$ is $O(g(x))$ and $f(x)$ is $\Omega(g(x))$.

EXAMPLE 11 Show that $3x^2 + 8x \log x$ is $\Theta(x^2)$.

Solution: Since $0 \leq 8x \log x \leq 8x^2$, it follows that $3x^2 + 8x \log x \leq 11x^2$ for $x \geq 1$. Consequently, $3x^2 + 8x \log x$ is $O(x^2)$. Clearly, x^2 is $O(3x^2 + 8x \log x)$. Consequently, $3x^2 + 8x \log x$ is $\Theta(x^2)$. ■

One useful fact is that the leading term of a polynomial determines its order. For example, if $f(x) = 3x^5 + x^4 + 17x^3 + 2$, then $f(x)$ is of order x^5. This is stated in the following theorem, whose proof is left as an exercise at the end of this section.

THEOREM 4 **Let $f(x) = a_nx^n + a_{n-1}x^{n-1} + \cdots + a_1x + a_0$, where $a_0, a_1, \ldots, a_n$ are real numbers with $a_n \neq 0$. Then $f(x)$ is of order x^n.**

EXAMPLE 12 The polynomials $3x^8 + 10x^7 + 221x^2 + 1444$, $x^{19} - 18x^4 - 10112$, and $-x^{99} + 40001x^{98} + 100003x$ are of orders x^8, x^{19}, and x^{99}, respectively. ■

Unfortunately, as Knuth observed, big-O notation is often used by careless writers and speakers as if it had the same meaning as big-Theta notation. Keep this in mind when you see big-O notation used. The recent trend has been to use big-Theta notation whenever both upper and lower bounds on the size of a function are needed.

Exercises

1. Determine whether each of the following functions is $O(x)$.

a) $f(x) = 10$ **b)** $f(x) = 3x + 7$
c) $f(x) = x^2 + x + 1$ **d)** $f(x) = 5 \log x$
e) $f(x) = \lfloor x \rfloor$ **f)** $f(x) = \lceil x/2 \rceil$

2. Determine whether each of the following functions is $O(x^2)$.

a) $f(x) = 17x + 11$ **b)** $f(x) = x^2 + 1000$
c) $f(x) = x \log x$ **d)** $f(x) = x^4/2$
e) $f(x) = 2^x$ **f)** $f(x) = \lfloor x \rfloor \cdot \lceil x \rceil$

3. Use the definition of the fact that $f(x)$ is $O(g(x))$ to show that $x^4 + 9x^3 + 4x + 7$ is $O(x^4)$.

4. Use the definition of the fact that $f(x)$ is $O(g(x))$ to show that $2^x + 17$ is $O(3^x)$.

5. Show that $(x^2 + 1)/(x + 1)$ is $O(x)$.

6. Show that $(x^3 + 2x)/(2x + 1)$ is $O(x^2)$.

7. Find the least integer n such that $f(x)$ is $O(x^n)$ for each of the following functions.

a) $f(x) = 2x^3 + x^2 \log x$
b) $f(x) = 3x^3 + (\log x)^4$
c) $f(x) = (x^4 + x^2 + 1)/(x^3 + 1)$
d) $f(x) = (x^4 + 5 \log x)/(x^4 + 1)$

8. Find the least integer n such that $f(x)$ is $O(x^n)$ for each of the following functions.

a) $f(x) = 2x^2 + x^3 \log x$
b) $f(x) = 3x^5 + (\log x)^4$
c) $f(x) = (x^4 + x^2 + 1)/(x^4 + 1)$
d) $f(x) = (x^3 + 5 \log x)/(x^4 + 1)$

9. Show that $x^2 + 4x + 17$ is $O(x^3)$ but that x^3 is not $O(x^2 + 4x + 17)$.

10. Show that x^3 is $O(x^4)$ but that x^4 is not $O(x^3)$.

11. Show that $3x^4 + 1$ is $O(x^4/2)$ and $x^4/2$ is $O(3x^4 + 1)$.

12. Show that $x \log x$ is $O(x^2)$ but that x^2 is not $O(x \log x)$.

13. Show that 2^n is $O(3^n)$ but that 3^n is not $O(2^n)$.

14. Is it true that x^3 is $O(g(x))$, if g is the given function? [For example, if $g(x) = x + 1$, this question asks whether x^3 is $O(x + 1)$.]

a) $g(x) = x^2$
b) $g(x) = x^3$
c) $g(x) = x^2 + x^3$
d) $g(x) = x^2 + x^4$
e) $g(x) = 3^x$
f) $g(x) = x^3/2$

15. Explain what it means for a function to be $O(1)$.

16. Show that if $f(x)$ is $O(x)$, then $f(x)$ is $O(x^2)$.

17. Suppose that $f(x)$, $g(x)$, and $h(x)$ are functions such that $f(x)$ is $O(g(x))$ and $g(x)$ is $O(h(x))$. Show that $f(x)$ is $O(h(x))$.

18. Let k be a positive integer. Show that $1^k + 2^k + \cdots + n^k$ is $O(n^{k+1})$.

19. Give as good a big-O estimate as possible for each of the following functions.
  a) $(n^2 + 8)(n + 1)$
  b) $(n \log n + n^2)(n^3 + 2)$
  c) $(n! + 2^n)(n^3 + \log(n^2 + 1))$

20. Give a big-O estimate for each of the following functions. For the function g in your estimate $f(x)$ is $O(g)$, use a simple function g of smallest order.
  a) $(n^3 + n^2 \log n)(\log n + 1) + (17 \log n + 19)(n^3 + 2)$
  b) $(2^n + n^2)(n^3 + 3^n)$
  c) $(n^n + n2^n + 5^n)(n! + 5^n)$

21. Give a big-O estimate for each of the following functions. For the function g in your estimate that $f(x)$ is $O(g(x))$ use a simple function g of the smallest order.
  a) $n \log(n^2 + 1) + n^2 \log n$
  b) $(n \log n + 1)^2 + (\log n + 1)(n^2 + 1)$
  c) $n^{2^n} + n^{n^2}$

22. For each function in Exercise 1, determine whether that function is $\Omega(x)$ and whether it is $\Theta(x)$.

23. For each function in Exercise 2, determine whether that function is $\Omega(x^2)$ and whether it is $\Theta(x^2)$.

24. **a)** Show that $3x + 7$ is $\Theta(x)$.
  b) Show that $2x^2 + x - 7$ is $\Theta(x^2)$.
  c) Show that $\lfloor x + 1/2 \rfloor$ is $\Theta(x)$.
  d) Show that $\log(x^2 + 1)$ is $\Theta(\log_2 x)$.
  e) Show that $\log_{10} x$ is $\Theta(\log_2 x)$.

25. Show that $f(x)$ is $\Theta(g(x))$ if and only if $f(x)$ is $O(g(x))$ and $g(x)$ is $O(f(x))$.

26. Show that if $f(x)$ and $g(x)$ are functions from the set of real numbers to the set of real numbers, then $f(x)$ is $O(g(x))$ if and only if $g(x)$ is $\Omega(f(x))$.

27. Show that if $f(x)$ and $g(x)$ are functions from the set of real numbers to the set of real numbers, then $f(x)$ is $\Theta(g(x))$ if and only if there are positive constants k, C_1, and C_2 such that $C_1|g(x)| \le |f(x)| \le C_2|g(x)|$ whenever $x > k$.

28. **a)** Show that $3x^2 + x + 1$ is $\Theta(3x^2)$ by directly finding the constants k, C_1, C_2 in Exercise 27.
  b) Express the relationship in part (a) using a picture showing the functions $3x^2 + x + 1$, $C_1 \cdot 3x^2$, and $C_2 \cdot 3x^2$, and the constant k on the x-axis, where C_1, C_2, and k are the constants you found in part (a) to show that $3x^2 + x + 1$ is $\Theta(3x^2)$.

29. Express the relationship $f(x)$ is $\Theta(g(x))$ using a picture. Show the graphs of the functions $f(x)$, $C_1|g(x)|$, and $C_2|g(x)|$, as well as the constant k on the x-axis.

30. Explain what it means for a function to be $\Omega(1)$.

31. Explain what it means for a function to be $\Theta(1)$.

32. Give a big-O estimate of the product of the first n odd positive integers.

33. Show that if f and g are real-valued functions such that $f(x)$ is $O(g(x))$, then $f^k(x)$ is $O(g^k(x))$. [Note that $f^k(x) = f(x)^k$.]

34. Show that if $f(x)$ is $O(\log_b x)$ where $b > 1$, then $f(x)$ is $O(\log_a x)$ where $a > 1$.

35. Suppose that $f(x)$ is $O(g(x))$ where f and g are increasing and unbounded functions. Show that $\log |f(x)|$ is $O(\log |g(x)|)$.

36. Suppose that $f(x)$ is $O(g(x))$. Does it follow that $2^{f(x)}$ is $O(2^{g(x)})$?

37. Let $f_1(x)$ and $f_2(x)$ be functions from the set of real numbers to the set of positive real numbers. Show that if $f_1(x)$ and $f_2(x)$ are both $\Theta(g(x))$, where $g(x)$ is a function from the set of real numbers to the set of positive real numbers, then $f_1(x) + f_2(x)$ is $\Theta(g(x))$. Is this still true if $f_1(x)$ and $f_2(x)$ can take negative values?

38. Suppose that $f(x)$, $g(x)$, and $h(x)$ are functions such that $f(x)$ is $\Theta(g(x))$ and $g(x)$ is $\Theta(h(x))$. Show that $f(x)$ is $\Theta(h(x))$.

39. If $f_1(x)$ and $f_2(x)$ are functions from the set of positive integers to the set of positive real numbers and $f_1(x)$ and $f_2(x)$ are both $\Theta(g(x))$, is $(f_1 - f_2)(x)$ also $\Theta(g(x))$? Either prove that it is or give a counterexample.

40. Show that if $f_1(x)$ and $f_2(x)$ are functions from the set of positive integers to the set of real numbers and $f_1(x)$ is $\Theta(g_1(x))$ and $f_2(x)$ is $\Theta(g_2(x))$, then $(f_1 f_2)(x)$ is $\Theta(g_1 g_2(x))$.

41. Find functions f and g from the set of positive integers to the set of real numbers such that $f(n)$ is not $O(g(n))$ and $g(n)$ is not $O(f(n))$ simultaneously.

42. Express the relationship $f(x)$ is $\Omega(g(x))$ using a picture. Show the graphs of the functions $f(x)$ and $Cg(x)$, as well as the constant k on the real axis.

43. Show that if $f_1(x)$ is $\Theta(g_1(x))$, $f_2(x)$ is $\Theta(g_2(x))$, and $f_2(x) \neq 0$ and $g_2(x) \neq 0$ for all real numbers $x > 0$, then $(f_1/f_2)(x)$ is $\Theta((g_1/g_2)(x))$.

44. Show that if $f(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$, where $a_0, a_1, \ldots, a_{n-1}, a_n$ are real numbers and $a_n \neq 0$, then $f(x)$ is $\Theta(x^n)$.

Big-O, big-Theta, and big-Omega notation can be extended to functions in more than one variable. For example, the statement $f(x, y)$ is $O(g(x, y))$ means that there exist constants C, k_1, and k_2 such that $|f(x, y)| \le C|g(x, y)|$ whenever $x > k_1$ and $y > k_2$.

45. Define the statement $f(x, y)$ is $\Theta(g(x, y))$.

46. Define the statement $f(x, y)$ is $\Omega(g(x, y))$.

47. Show that $(x^2 + xy + x \log y)^3$ is $O(x^6 y^3)$.

48. Show that $x^5 y^3 + x^4 y^4 + x^3 y^5$ is $\Omega(x^3 y^3)$.

49. Show that $\lfloor xy \rfloor$ is $O(xy)$.

50. Show that $\lceil xy \rceil$ is $\Omega(xy)$.

The following problems deal with another type of asymptotic notation, called **little-o** notation. Because little-o

notation is based on the concept of limits, a knowledge of calculus is needed for these problems. We say that $f(x)$ is $o(g(x))$ [read $f(x)$ is "little-oh" of $g(x)$], when

$$\lim_{x \to \infty} \frac{f(x)}{g(x)} = 0.$$

51. (Calculus required) Show that

a) x^2 is $o(x^3)$ **b)** $x \log x$ is $o(x^2)$

c) x^2 is $o(2^x)$ **d)** $x^2 + x + 1$ is not $o(x^2)$

52. (Calculus required)

a) Show that if $f(x)$ and $g(x)$ are functions such that $f(x)$ is $o(g(x))$ and c is a constant, then $cf(x)$ is $o(g(x))$ where $(cf)(x) = cf(x)$.

b) Show that if $f_1(x)$, $f_2(x)$, and $g(x)$ are functions such that $f_1(x)$ is $o(g(x))$ and $f_2(x)$ is $o(g(x))$, then $(f_1 + f_2)(x)$ is $o(g(x))$, where $(f_1 + f_2)(x) = f_1(x) + f_2(x)$.

53. (Calculus required) Represent pictorially that $x \log x$ is $o(x^2)$ by graphing $x \log x$, x^2, and $x \log x / x^2$. Explain how this picture shows that $x \log x$ is $o(x^2)$.

54. (Calculus required) Express the relationship $f(x)$ is $o(g(x))$ using a picture. Show the graphs of $f(x)$, $g(x)$, and $f(x)/g(x)$.

***55.** (Calculus required) Suppose that $f(x)$ is $o(g(x))$. Does it follow that $2^{f(x)}$ is $o(2^{g(x)})$?

***56.** (Calculus required) Suppose that $f(x)$ is $o(g(x))$. Does it follow that $\log |f(x)|$ is $o(\log |g(x)|)$?

57. (Calculus required) The two parts of this exercise describe the relationship between little-o and big-O notation.

a) Show that if $f(x)$ and $g(x)$ are functions such that $f(x)$ is $o(g(x))$, then $f(x)$ is $O(g(x))$.

b) Show that if $f(x)$ and $g(x)$ are functions such that $f(x)$ is $O(g(x))$, then it does not necessarily follow that $f(x)$ is $o(g(x))$.

58. (Calculus required) Show that if $f(x)$ is a polynomial of degree n and $g(x)$ is a polynomial of degree m where $m > n$, then $f(x)$ is $o(g(x))$.

59. (Calculus required) Show that if $f_1(x)$ is $O(g(x))$ and $f_2(x)$ is $o(g(x))$, then $f_1(x) + f_2(x)$ is $O(g(x))$.

60. (Calculus required) Let H_n be the nth **harmonic number**

$$H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}.$$

Show that H_n is $O(\log n)$. (*Hint:* First establish the inequality

$$\sum_{j=2}^{n} \frac{1}{j} < \int_1^n \frac{1}{x}\,dx$$

by showing that the sum of the areas of the rectangles of height $1/j$ with base from $j - 1$ to j, for $j = 2, 3, \ldots, n$, is less than the area under the curve $y = 1/x$ from 2 to n.)

***61.** Show that $n \log n$ is $O(\log n!)$.

62. Determine whether $\log(n!)$ is $O(n \log n)$. Justify your answer.

Let $f(x)$ and $g(x)$ be functions from the set of real numbers to the set of real numbers. We say that f and g are **asymptotic** and write $f(x) \sim g(x)$ if $\lim_{x \to \infty} f(x)/g(x) = 1$.

63. (Calculus required) For each of the following pairs of functions, determine whether f and g are asymptotic.

a) $f(x) = x^2 + 3x + 7$, $g(x) = x^2 + 10$

b) $f(x) = x^2 \log x$, $g(x) = x^3$

c) $f(x) = x^4 + \log(3x^8 + 7)$, $g(x) = (x^2 + 17x + 3)^2$

d) $f(x) = (x^3 + x^2 + x + 1)^4$, $g(x) = (x^4 + x^3 + x^2 + x + 1)^3$.

e) $f(x) = \log(x^2 + 1)$, $g(x) = \log x$

f) $f(x) = 2^{x+3}$, $g(x) = 2^{x+7}$

g) $f(x) = 2^{2^x}$, $g(x) = 2^{x^2}$

Key Terms and Results

LOGIC (SECTIONS 1–3):

TERMS

proposition: a statement that is true or false

truth value: true or false

$\neg p$ (negation of p): the proposition with truth value opposite to the truth value of p

logical operators: operators used to combine propositions

compound proposition: a proposition constructed by combining propositions using logical operators

truth table: a table displaying the truth values of propositions

$p \vee q$ (disjunction of p and q): the proposition that is true unless both p and q are false

$p \wedge q$ (conjunction of p and q): the proposition that is true only when both p and q are true

$p \oplus q$ (exclusive or of p and q): the proposition that is true when exactly one of p and q is true

$p \to q$ (p implies q): the proposition that is false only when p is true and q is false

converse of $p \to q$: the implication $q \to p$

contrapositive of $p \to q$: the implication $\neg q \to \neg p$

$p \leftrightarrow q$ (biconditional): the proposition that is true only when p and q have the same truth value

bit: either a 0 or a 1

Boolean variable: a variable that has a value of 0 or 1

bit operation: an operation on a bit or bits

bit string: a list of bits

bitwise operations: operations on bit strings that operate on each bit in one string and the corresponding bit in the other string

tautology: a compound proposition that is always true

contradiction: a compound proposition that is always false

contingency: a compound proposition that is sometimes true and sometimes false

logical equivalence: compound propositions are logically equivalent if they always have the same truth values

propositional function: the combination of a variable and a predicate

universe of discourse: the domain of the variable in a propositional function

$\exists x\, P(x)$ (existential quantification of $P(x)$): the proposition that is true if and only if there exists an x in the universe of discourse such that $P(x)$ is true

$\forall x\, P(x)$ (universal quantification of $P(x)$): the proposition that is true if and only if $P(x)$ is true for all x in the universe of discourse

free variable: a variable not bound in a propositional function

RESULTS

The logical equivalences given in Tables 5 and 6 in Section 2

SETS (SECTIONS 4–5):

TERMS

set: a collection of distinct objects

axiom: a basic assumption of a theory

paradox: a logical inconsistency

element, member of a set: an object in a set

$\emptyset$ (empty set, null set): the set with no members

universal set: the set containing all objects under consideration

Venn diagram: a graphical representation of a set or sets

$S = T$ (set equality): S and T have the same elements

$S \subseteq T$ (S is a subset of T): every element of S is also an element of T

$S \subset T$ (S is a proper subset of T): S is a subset of T and $S \neq T$

finite set: a set with n elements where n is a nonnegative integer

infinite set: a set that is not finite

$|S|$ (the cardinality of S): the number of elements in S

$P(S)$ (the power set of S): the set of all subsets of S

$A \cup B$ (the union of A and B): the set containing those elements that are in at least one of A and B

$A \cap B$ (the intersection of A and B): the set containing those elements that are in both A and B.

$A - B$ (the difference of A and B): the set containing those elements that are in A but not in B

A (the complement of A): the set of elements in the universal set that are not in A

$A \oplus B$ (the symmetric difference of A and B): the set containing those elements in exactly one of A and B

membership table: a table displaying the membership of elements in sets

RESULTS

The set identities given in Table 1 in Section 5

FUNCTIONS (SECTIONS 6–8):

TERMS

function from A to B: an assignment of exactly one element of B to each element of A

domain of f: the set A where f is a function from A to B

codomain of f: the set B where f is a function from A to B

b is the image of a under f: $b = f(a)$

a is a pre-image of b under f: $f(a) = b$

range of f: the set of images of f

onto function, surjection: a function from A to B such that every element of B is the image of some element in A

one-to-one function, injection: a function such that the images of elements in its domain are all different

one-to-one correspondence, bijection: a function that is both one-to-one and onto

inverse of f: the function that reverses the correspondence given by f (when f is a bijection)

$f \circ g$ (composition of f and g): the function that assigns $f(g(x))$ to x

$\lfloor x \rfloor$ (floor function): the largest integer not exceeding x

$\lceil x \rceil$ (ceiling function): the smallest integer greater than or equal to x

sequence: a function with domain that is a subset of the set of integers

string: a finite sequence

$\sum_{i=1}^{n} a_i$: the sum $a_1 + a_2 + \cdots + a_n$

$\prod_{i=1}^{n} a_i$: the product $a_1 a_2 \cdots a_n$

countable set: a set that is either finite or that can be placed in one-to-one correspondence with the set of positive integers

uncountable set: a set that is *not* countable

$f(x)$ is $O(g(x))$: the fact that $|f(x)| \le C|g(x)|$ for all $x > k$ for some constants C and k

$f(x)$ is $\Omega(g(x))$: the fact that $|f(x)| \ge C|g(x)|$ for all $x > k$ for some positive constants C and k

$f(x)$ is $\Theta(g(x))$: the fact that $f(x)$ is $O(g(x))$ and $f(x)$ is $\Omega(g(x))$

RESULTS

The set of real numbers is uncountable.

$\log n!$ is $O(n \log n)$.

If $f_1(x)$ is $O(g_1(x))$ and $f_2(x)$ is $O(g_2(x))$, then $(f_1 + f_2)(x)$ is $O(\max(g_1(x), g_2(x)))$ and $(f_1 f_2)(x)$ is $O(g_1(x) g_2(x))$.

If $a_0, a_1, \ldots, a_n$ are real numbers, then $a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$ is $O(x^n)$ and $\Theta(x^n)$.

Review Questions

1. a) Define the negation of a proposition.
   b) What is the negation of "This is a boring course"?

2. a) Define (using truth tables) the disjunction, conjunction, exclusive or, conditional, and biconditional of the propositions p and q.
   b) What are the disjunction, conjunction, exclusive or, implication, and biconditional of the propositions "I'll go to the movies tonight" and "I'll finish my discrete mathematics homework"?

3. a) Describe at least five different ways to write the implication $p \rightarrow q$ in English.
   b) Define the converse and contrapositive of an implication.
   c) State the converse and the contrapositive of the implication "If it is sunny tomorrow, then I will go for a walk in the woods."

4. a) What does it mean for two propositions to be logically equivalent?
   b) Describe the different ways to show that two compound propositions are logically equivalent.
   c) Show in at least two different ways that the compound propositions $\neg p \vee (r \rightarrow \neg q)$ and $\neg p \vee \neg q \vee \neg r$ are equivalent.

5. *(Depends on the Exercise Set in Section 1.2)*
   a) Given a truth table, explain how to use disjunctive normal form to construct a compound proposition with this truth table.
   b) Explain why part (a) shows that the operators $\wedge$, $\vee$, and $\neg$ are functionally complete.
   c) Is there an operator such that the set containing just this operator is functionally complete?

6. What are the universal and existential quantifications of a predicate $P(x)$? What are their negations?

7. a) What is the difference between the quantification $\exists x \forall y P(x, y)$ and $\forall y \exists x P(x, y)$, where $P(x, y)$ is a predicate?
   b) Give an example of a predicate $P(x, y)$ such that $\exists x \forall y P(x, y)$ and $\forall y \exists x P(x, y)$ have different truth values.

8. a) Define the union, intersection, difference, and symmetric difference of two sets.
   b) What are the union, intersection, difference, and symmetric difference of the set of positive integers and the set of odd integers?

9. a) Define what it means for two sets to be equal.
   b) Describe the ways to show that two sets are equal.
   c) Show in at least two different ways that the sets $A - (B \cap C)$ and $(A - B) \cup (A - C)$ are equal.

10. Explain the relationship between logical equivalences and set identities.

11. a) Define $|S|$, the cardinality of the set S.
   b) Give a formula for $|A \cup B|$ where A and B are sets.

12. a) Define the power set of a set S.
   b) When is the empty set in the power set of a set S?
   c) How many elements does the power set of a set S with n elements have?

13. **a)** Define the domain, codomain, and the range of a function.
  b) Let $f(n)$ be the function from the set of integers to the set of integers such that $f(n) = n^2 + 1$. What are the domain, codomain, and range of this function?

14. **a)** Define what it means for a function from the set of positive integers to the set of positive integers to be one-to-one.
  b) Define what it means for a function from the set of positive integers to the set of positive integers to be onto.
  c) Give an example of a function from the set of positive integers to the set of positive integers that is both one-to-one and onto.
  d) Give an example of a function from the set of positive integers to the set of positive integers that is one-to-one but not onto.
  e) Give an example of a function from the set of positive integers to the set of positive integers that is not one-to-one but is onto.
  f) Give an example of a function from the set of positive integers to the set of positive integers that is neither one-to-one nor onto.

15. **a)** Define the inverse of a function.
  b) When does a function have an inverse?
  c) Does the function $f(n) = 10 - n$ from the set of integers to the set of integers have an inverse? If so, what is it?

16. **a)** Define the floor and ceiling functions from the set of real numbers to the set of integers.
  b) For which real numbers x is it true that $\lfloor x \rfloor = \lceil x \rceil$?

17. **a)** Use summation notation to express the sum of the powers of 2 from 2^0 to 2^n.
  b) What is the value of the sum in part (a)?

18. **a)** What does it mean for a set to be countable? Give a precise definition.
  b) Is the set of negative integers countable? Why or why not?
  c) Is the set of rational numbers with denominators greater than 3 countable? Why or why not?
  d) Is the set of real numbers between 2 and 3 countable? Why or why not?

19. **a)** State the definition of the fact that $f(n)$ is $O(g(n))$, where $f(n)$ and $g(n)$ are functions from the set of positive integers to the set of real numbers.
  b) Use the definition of the fact that $f(n)$ is $O(g(n))$ directly to prove or disprove that $n^2 + 18n + 107$ is $O(n^3)$.
  c) Use the definition of the fact that $f(n)$ is $O(g(n))$ directly to prove or disprove that n^3 is $O(n^2 + 18n + 107)$.

20. **a)** How can you produce a big-O estimate for a function that is the sum of different terms where each term is the product of several functions?
  b) Give a big-O estimate for the function $f(n) = (n! + 1)(2^n + 1) + (n^{n-2} + 8n^{n-3})(n^3 + 2^n)$. For the function g in your estimate $f(x)$ is $O(g(x))$ use a simple function of smallest possible order.

Supplementary Exercises

1. Let p be the proposition "I will do every exercise in this book" and q be the proposition "I will get an 'A' in this course." Express each of the following as a combination of p and q.
  a) I will get an 'A' in this course only if I do every exercise in this book.
  b) I will get an 'A' in this course and I will do every exercise in this book.
  c) Either I will not get an 'A' in this course or I will not do every exercise in this book.
  d) For me to get an 'A' in this course it is necessary and sufficient that I do every exercise in this book.

2. Find the truth table of the compound proposition $(p \vee q) \rightarrow (p \wedge \neg r)$.

3. Show that the following propositions are tautologies.
  a) $(\neg q \wedge (p \rightarrow q)) \rightarrow \neg p$
  b) $((p \vee q) \wedge \neg p) \rightarrow q$

4. Give the converse and the contrapositive of the following implications.
  a) If it rains today, then I will drive to work.
  b) If $|x| = x$, then $x \geq 0$.
  c) If n is greater than 3, then n^2 is greater than 9.

5. Find a compound proposition involving the propositional variables p, q, r, and s that is true when exactly three of these propositional variables are true and is false otherwise.

6. Let $P(x)$ be the statement "student x knows calculus" and let $Q(y)$ be the statement "class y contains a student who knows calculus." Express each of the following as quantifications of $P(x)$ and $Q(y)$.

a) Some students know calculus.
b) Not every student knows calculus.
c) Every class has a student in it who knows calculus.
d) Every student in every class knows calculus.
e) There is at least one class with no students who know calculus.

7. Let $P(m, n)$ be the statement "m divides n," where the universe of discourse for both variables is the set of positive integers. Determine the truth values of each of the following propositions.

a) $P(4, 5)$
b) $P(2, 4)$
c) $\forall m \forall n\, P(m, n)$
d) $\exists m \forall n\, P(m, n)$
e) $\exists n \forall m\, P(m, n)$
f) $\forall n\, P(1, n)$

8. Let $P(x, y)$ be a propositional function. Show that the implication $\exists x \forall y\, P(x, y) \rightarrow \forall y \exists x\, P(x, y)$ is a tautology.

9. Let $P(x)$ and $Q(x)$ be propositional functions. Show that $\exists x\,(P(x) \rightarrow Q(x))$ and $\forall x\, P(x) \rightarrow \exists x\, Q(x)$ always have the same truth value.

10. If $\forall y \exists x\, P(x, y)$ is true, does it necessarily follow that $\exists x \forall y\, P(x, y)$ is true?

11. If $\forall x \exists y\, P(x, y)$ is true, does it necessarily follow that $\exists x \forall y\, P(x, y)$ is true?

12. Find the negations of the following statements.

a) If it snows today, then I will go skiing tomorrow.
b) Every person in this class understands mathematical induction.
c) Some students in this class do not like discrete mathematics.
d) In every mathematics class there is some student who falls asleep during lectures.

13. Express the following statement using quantifiers: "Every student in this class has taken some course in every department in the school of mathematical sciences."

14. Express the following statement using quantifiers: "There is a building on the campus of some college in the United States in which every room is painted white."

15. Let A be the set of English words that contain the letter x, and let B be the set of English words that contain the letter q. Express each of the following sets as a combination of A and B.

a) The set of English words that do not contain the letter x.
b) The set of English words that contain both an x and a q.
c) The set of English words that contain an x but not a q.
d) The set of English words that do not contain either an x or a q.
e) The set of English words that contain an x or a q, but not both.

16. Show that if A is a subset of B, then the power set of A is a subset of the power set of B.

17. Suppose that A and B are sets such that the power set of A is a subset of the power set of B. Does it follow that A is a subset of B?

18. Let $\mathbf{E}$ denote the set of even integers and $\mathbf{O}$ denote the set of odd integers. As usual, let $\mathbf{Z}$ denote the set of all integers. Determine each of the following.

a) $\mathbf{E} \cup \mathbf{O}$
b) $\mathbf{E} \cap \mathbf{O}$
c) $\mathbf{Z} - \mathbf{E}$
d) $\mathbf{Z} - \mathbf{O}$

19. Show that if A is a set and U is the universal set, then

a) $A \cap \bar{A} = \emptyset$.
b) $A \cup \bar{A} = U$.

20. Show that if A and B are sets, then

a) $A = A \cap (A \cup B)$.
b) $A = A \cup (A \cap B)$.

21. Show that if A and B are sets, then $A - (A - B) = A \cap B$.

22. Let A and B be sets. Show that $A \subseteq B$ if and only if $A \cap B = A$.

23. Let A, B, and C be sets. Show that $(A - B) - C$ is not necessarily equal to $A - (B - C)$.

24. Suppose that A, B, and C are sets. Prove or disprove that $(A - B) - C = (A - C) - B$.

25. Suppose that A, B, C, and D are sets. Prove or disprove that $(A - B) - (C - D) = (A - C) - (B - D)$.

26. Show that if A and B are finite sets, then $|A \cap B| \leq |A \cup B|$. Determine when this relationship is an equality.

27. Let A and B be sets in a finite universal set U. List the following in order of increasing size.

a) $|A|, |A \cup B|, |A \cap B|, |U|, |\emptyset|$
b) $|A - B|, |A \oplus B|, |A| + |B|, |A \cup B|, |\emptyset|$

28. Let A and B be subsets of the finite universal set U. Show that $|\bar{A} \cap \bar{B}| = |U| - |A| - |B| + |A \cap B|$.

29. Let f and g be functions from $\{1, 2, 3, 4\}$ to $\{a, b, c, d\}$ and from $\{a, b, c, d\}$ to $\{1, 2, 3, 4\}$, respectively, such that $f(1) = d$, $f(2) = c$, $f(3) = a$, $f(4) = b$, and $g(a) = 2$, $g(b) = 1$, $g(c) = 3$, $g(d) = 2$.

a) Is f one-to-one? Is g one-to-one?
b) Is f onto? Is g onto?
c) Does either f or g have an inverse? If so, find this inverse.

30. Let f be a one-to-one function from the set A to the set B. Let S and T be subsets of A. Show that $f(S \cap T) = f(S) \cap f(T)$.

31. Give an example to show that the equality in Exercise 30 may not hold if f is not one-to-one.

32. Show that if n is an integer, then $n = \lceil n/2 \rceil + \lfloor n/2 \rfloor$.

33. Find the value of the following quantities.

a) $\displaystyle\sum_{i=0}^{3}\left(\sum_{j=0}^{4} ij\right)$
b) $\displaystyle\prod_{i=1}^{4}\left(\sum_{i=0}^{3} j\right)$

c) $\displaystyle\sum_{i=1}^{5}\left(\sum_{j=0}^{i} 1\right)$
d) $\displaystyle\prod_{i=1}^{3}\left(\prod_{j=0}^{i} j\right)$

34. Is the set of irrational numbers between 0 and 1 countable? Justify your answer.

****35.** A real number is called **algebraic** if it is the root of a polynomial with integer coefficients. Show that there are a countable number of algebraic numbers. (*Hint:* Use the fact that a polynomial of degree n has at most n distinct roots.)

36. Show that $8x^3 + 12x + 100 \log x$ is $O(x^3)$.

37. Give a big-O estimate for $(x^2 + x(\log x)^3) \cdot (2^x + x^3)$.

38. Find a big-O estimate for $\sum_{j=1}^{n} j(j+1)$.

***39.** Show that $n!$ is not $O(2^n)$.

***40.** Show that n^n is not $O(n!)$.

Computer Projects

WRITE PROGRAMS WITH THE SPECIFIED INPUT AND OUTPUT.

1. Given the truth values of the propositions p and q, find the truth values of the conjunction, disjunction, exclusive or, implication, and biconditional of these propositions.
2. Given two bit strings of length n, find the bitwise *AND*, bitwise *OR*, and bitwise *XOR* of these strings.
3. Given the truth values of the propositions p and q in fuzzy logic, find the truth value of the disjunction and the conjunction of p and q (see Exercises 31–33 of Section 1.1).
4. Given subsets A and B of a set with n elements, use bit strings to find $\overline{A}$, $A \cup B$, $A \cap B$, $A - B$, and $A \oplus B$.
5. Given multisets A and B from the same universal set, find $A \cup B$, $A \cap B$, $A - B$, and $A + B$ (see preamble to Exercise 47 of Section 1.5).
6. Given fuzzy sets A and B, find $\overline{A}$, $A \cup B$, and $A \cap B$ (see preamble to Exercise 49 of Section 1.5).
7. Given a function f from $\{1, 2, \ldots, n\}$ to the set of integers, determine whether f is one-to-one.
8. Given a function f from $\{1, 2, \ldots, n\}$ to itself, determine whether f is onto.
9. Given a bijection f from the set $\{1, 2, \ldots, n\}$ to itself, find f^{-1}.
10. Given the terms of a sequence $a_1, a_2, \ldots, a_n$, find $\sum_{j=1}^{n} a_j$ and $\prod_{j=1}^{n} a_j$.

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. What is the largest value of n for which $n!$ has fewer than 100 decimal digits and fewer than 1000 decimal digits?
2. How many zeros are there at the end of the decimal representation of $n!$ for each of the first 25 positive integers n? Can you figure out a formula for the number of zeros at the end of $n!$ in decimal notation? (See Section 2.3.)
3. Calculate the number of one-to-one functions from a set S to a set T, where S and T are finite sets of various sizes. Can you determine a formula for the number of such functions? (We will find such a formula in Chapter 4.)
4. Calculate the number of onto functions from a set S to a set T where S and T are finite sets of various sizes. Can you determine a formula for the number of such functions? (We will find such a formula in Chapter 5.)
5. We know that n^b is $O(d^n)$ when b and d are positive numbers with $d \geq 2$. Give values of the constants C and k such that $n^b \leq Cd^n$ whenever $x > k$ for each of the following sets of values: $b = 10, d = 2$; $b = 20, d = 3$; $b = 1000, d = 7$.

Writing Projects

RESPOND TO THE FOLLOWING WITH ESSAYS USING OUTSIDE SOURCES.

1. Describe how fuzzy logic is being applied to practical applications. Consult one or more of the recent books on fuzzy logic written for general audiences.
2. Read some of the writings of Lewis Carroll on symbolic logic. Describe in detail some of the models he used to represent logical arguments.

3. Discuss how an axiomatic set theory can be developed to avoid Russell's paradox. (See Exercise 26 of Section 1.4.)
4. Research where the concept of a function first arose, and describe how this concept was first used.
5. Explain how various people have found *The Encyclopedia of Integer Sequences* [SIPl95] useful. Also, describe a few of the more unusual sequences in this encyclopedia and how they arise.
6. Describe how the concept of the cardinality of sets can be extended to infinite sets.
7. Look up the definition of a transcendental number. Explain how to show that such numbers exist and how such numbers can be constructed. Which famous numbers can be shown to be transcendental?
8. Look up Bachmann's original introduction of big-O notation. Explain how he and others have used this notation.

The Fundamentals: Algorithms, the Integers, and Matrices

2

Many problems can be solved by considering them as special cases of general problems. For instance, consider the problem of locating the largest integer in the sequence 101, 12, 144, 212, 98. This is a specific case of the problem of locating the largest integer in a sequence of integers. To solve this general problem we must give an algorithm, which specifies a sequence of steps used to solve this general problem. We will study algorithms for solving many different types of problems in this book. For instance, algorithms will be developed for finding the greatest common divisor of two integers, for generating all the orderings of a finite set, for searching a list, and for finding the shortest path between two vertices in a network. One important consideration concerning an algorithm is its computational complexity. That is, what are the computer resources needed to use this algorithm to solve a problem of a specified size? We will illustrate the analysis of the complexity of algorithms in this chapter.

The set of integers plays a fundamental role in discrete mathematics. In particular, the concept of division of integers is fundamental to computer arithmetic. We will briefly review some of the important concepts of number theory, the study of integers and their properties. Some important algorithms involving integers will be studied, including the Euclidean algorithm for computing greatest common divisors, which was first described thousands of years ago. Integers can be represented using any positive integer greater than 1 as a base. Binary expansions, which are used throughout computer science, are representations with 2 as the base. In this chapter we discuss base b representations of integers and give an algorithm for finding them. Algorithms for integer arithmetic, which were the first procedures called algorithms, will also be discussed. This chapter also introduces several important applications of number theory. For example, in this chapter we will use number theory to make messages secret, to generate pseudorandom numbers, and to assign memory locations to computer files. Number theory, once considered the purest of subjects, has become an essential tool in providing computer and Internet security.

Matrices are used in discrete mathematics to represent a variety of discrete structures. We review the basic material about matrices and matrix arithmetic needed to represent relations and graphs. Matrix arithmetic will be used in numerous algorithms involving these structures.

2.1

Algorithms

INTRODUCTION

There are many general classes of problems that arise in discrete mathematics. For instance: given a sequence of integers, find the largest one; given a set, list all of its subsets; given a set of integers, put them in increasing order; given a network, find the shortest path between two vertices. When presented with such a problem, the first thing to do is to construct a model that translates the problem into a mathematical context. Discrete structures used in such models include sets, sequences, and functions—structures discussed in Chapter 1—as well as such other structures as permutations, relations, graphs, trees, networks, and finite state machines—concepts that will be discussed in later chapters.

Setting up the appropriate mathematical model is only part of the solution. To complete the solution, a method is needed that will solve the general problem using the model. Ideally, what is required is a procedure that follows a sequence of steps that leads to the desired answer. Such a sequence of steps is called an **algorithm.**

DEFINITION 1. **An *algorithm* is a finite set of precise instructions for performing a computation or for solving a problem.**

The term *algorithm* is a corruption of the name *al-Khowarizmi*, an Arabian mathematician of the ninth century, whose book on Hindu numerals is the basis of modern decimal notation. Originally, the word *algorism* was used for the rules for performing arithmetic using decimal notation. *Algorism* evolved into the word *algorithm* by the eighteenth century. With the growing interest in computing machines, the concept of an algorithm was given a more general meaning, to include all definite procedures for solving problems, not just the procedures for performing arithmetic. (We will discuss algorithms for performing arithmetic with integers in Section 2.4.)

In this book, we will discuss algorithms that solve a wide variety of problems. In this section we will use the problem of finding the largest integer in a finite sequence of integers to illustrate the concept of an algorithm and the properties algorithms have. Also, we will describe algorithms for locating a particular element in a finite set. In subsequent sections, procedures for finding the greatest common divisor of two integers, for finding the shortest path between two points in a network, for multiplying matrices, and so on, will be discussed.

EXAMPLE 1 Describe an algorithm for finding the maximum (largest) value in a finite sequence of integers.

Even though the problem of finding the maximum element in a sequence is relatively trivial, it provides a good illustration of the concept of an algorithm. Also, there are many instances where the largest integer in a finite sequence of integers is required. For instance, a university may need to find the highest score on a competitive exam taken by thousands of students. Or a sports organization may want to identify the

---

web

Abu Ja'far Mohammed ibn Musa al-Khowarizmi (c. 780–c. 850). al-Khowarizmi, an astronomer and mathematician, was a member of the House of Wisdom, an academy of scientists in Baghdad. The name al-Khowarizmi means "from the town of Kowarzizm," which is now called *Khiva* and is part of Uzbekistan. al-Khowarizmi wrote books on mathematics, astronomy, and geography. Western Europeans first learned about algebra from his works. The word *algebra* comes from al-jabr, part of the title of his book *Kitab al-jabr w'al muqabala*. This book was translated into Latin and was a widely used textbook. His book on the use of Hindu numerals describes procedures for arithmetic operations using these numerals. European authors used a Latin corruption of his name, which later evolved to the word *algorithm* to describe the subject of arithmetic with Hindu numerals.

member with the highest rating each month. We want to develop an algorithm that can be used whenever the problem of finding the largest element in a finite sequence of integers arises.

We can specify a procedure for solving this problem in several ways. One method is simply to use the English language to describe the sequence of steps used. We now provide such a solution.

Solution of Example 1: We perform the following steps.

1. Set the temporary maximum equal to the first integer in the sequence. (The temporary maximum will be the largest integer examined at any stage of the procedure.)
2. Compare the next integer in the sequence to the temporary maximum, and if it is larger than the temporary maximum, set the temporary maximum equal to this integer.
3. Repeat the previous step if there are more integers in the sequence.
4. Stop when there are no integers left in the sequence. The temporary maximum at this point is the largest integer in the sequence. ■

An algorithm can also be described using a computer language. However, when that is done, only those instructions permitted in the language can be used. This often leads to a description of the algorithm that is complicated and difficult to understand. Furthermore, since many programming languages are in common use, it would be undesirable to choose one particular language. So, instead of using a particular computer language to specify algorithms, a form of **pseudocode** will be used in this book. (All algorithms will also be described using the English language.) Pseudocode provides an intermediate step between an English language description of an algorithm and an implementation of this algorithm in a programming language. The steps of the algorithm are specified using instructions resembling those used in programming languages. However, in pseudocode, the instructions used can include any well-defined operations or statements. A computer program can be produced in any computer language using the pseudocode description as a starting point.

The pseudocode used in this book is loosely based on the programming language Pascal. However, the syntax of Pascal, or that of other programming languages, will not be followed. Furthermore, any well-defined instruction can be used in this pseudocode. The details of the pseudocode used in the text are given in Appendix 2. The reader should refer to this appendix whenever the need arises.

A pseudocode description of the algorithm for finding the maximum element in a finite sequence follows.

ALGORITHM 1 **Finding the Maximum Element in a Finite Sequence.**

```
procedure max($a_1, a_2, \ldots, a_n$: integers)
$max := a_1$
for $i := 2$ to $n$
    if $max < a_i$ then $max := a_i$
{$max$ is the largest element}
```

This algorithm first assigns the initial term of the sequence, a_1, to the variable *max*. The "for" loop is used to successively examine terms of the sequence. If a term is greater than the current value of *max*, it is assigned to be the new value of *max*.

There are several properties that algorithms generally share. They are useful to keep in mind when algorithms are described. These properties are:

- *Input.* An algorithm has input values from a specified set.
- *Output.* From each set of input values an algorithm produces output values from a specified set. The output values are the solution to the problem.
- *Definiteness.* The steps of an algorithm must be defined precisely.
- *Correctness.* An algorithm should produce the correct output values for each set of input values.
- *Finiteness.* An algorithm should produce the desired output after a finite (but perhaps large) number of steps for any input in the set.
- *Effectiveness.* It must be possible to perform each step of an algorithm exactly and in a finite amount of time.
- *Generality.* The procedure should be applicable for all problems of the desired form, not just for a particular set of input values.

EXAMPLE 2 Show that Algorithm 1 for finding the maximum element in a finite sequence of integers has all the properties listed.

Solution: The input to Algorithm 1 is a sequence of integers. The output is the largest integer in the sequence. Each step of the algorithm is precisely defined, since only assignments, a finite loop, and conditional statements occur. The algorithm uses a finite number of steps, since it terminates after all the integers in the sequence have been examined. The algorithm can be carried out in a finite amount of time since each step is either a comparison or an assignment. Finally, Algorithm 1 is general, since it can be used to find the maximum of any finite sequence of integers. ■

SEARCHING ALGORITHMS

web

The problem of locating an element in an ordered list occurs in many contexts. For instance, a program that checks the spelling of words searches for them in a dictionary, which is just an ordered list of words. Problems of this kind are called **searching problems.** We will discuss several algorithms for searching in this section. We will study the number of steps used by each of these algorithms in Section 2.2.

The general searching problem can be described as follows: Locate an element x in a list of distinct elements $a_1, a_2, \ldots, a_n$, or determine that it is not in the list. The solution to this search problem is the location of the term in the list that equals x (that is, i is the solution if $x = a_i$) and is 0 if x is not in the list.

The first algorithm that we will present is called the **linear search,** or **sequential search,** algorithm. The linear search algorithm begins by comparing x and a_1. When $x = a_1$, the solution is the location of a_1, namely, 1. When $x \neq a_1$, compare x with a_2. If $x = a_2$, the solution is the location of a_2, namely, 2. When $x \neq a_2$, compare x with a_3. Continue this process, comparing x successively with each term of the list until a match is found, where the solution is the location of that term, unless no match occurs.

If the entire list has been searched without locating x, the solution is 0. The pseudocode for the linear search algorithm is displayed as Algorithm 2.

ALGORITHM 2 **The Linear Search Algorithm.**

```
procedure linear search($x$: integer, $a_1, a_2, \ldots, a_n$: distinct integers)
$i := 1$
while ($i \leq n$ and $x \neq a_i$)
    $i := i + 1$
if $i \leq n$ then location := $i$
else location := 0
{location is the subscript of term that equals $x$, or is 0 if $x$ is not
    found}
```

We will now consider another searching algorithm. This algorithm can be used when the list has terms occurring in order of increasing size (for instance: if the terms are numbers, they are listed from smallest to largest; if they are words, they are listed in lexicographic, or alphabetic, order). This second algorithm is called the **binary search algorithm.** It proceeds by comparing the element to be located to the middle term of the list. The list is then split into two smaller sublists of the same size, or where one of these smaller lists has one fewer term than the other. The search continues by restricting the search to the appropriate sublist based on the comparison of the element to be located and the middle term. In the next section, it will be shown that the binary search algorithm is much more efficient than the linear search algorithm. The following example demonstrates how a binary search works.

EXAMPLE 3 To search for 19 in the list

1 2 3 5 6 7 8 10 12 13 15 16 18 19 20 22.

first split this list, which has 16 terms, into two smaller lists with eight terms each, namely,

1 2 3 5 6 7 8 10   12 13 15 16 18 19 20 22.

Then, compare 19 and the largest term in the first list. Since $10 < 19$, the search for 19 can be restricted to the list containing the 9th through the 16th terms of the original list. Next, split this list, which has eight terms, into the two smaller lists of four terms each, namely,

12 13 15 16   18 19 20 22.

Since $16 < 19$ (comparing 19 with the largest term of the first list) the search is restricted to the second of these lists, which contains the 13th through the 16th terms of the original list. The list 18 19 20 22 is split into two lists, namely,

18 19   20 22.

Since 19 is not greater than the largest term of the first of these two lists, which is also 19, the search is restricted to the first list: 18 19, which contains the 13th and 14th terms of the original list. Next, this list of two terms is split into two lists of one term each: 18 and 19. Since $18 < 19$, the search is restricted to the second list: the list containing the 14th term of the list, which is 19. Now that the search has been narrowed down to one term, a comparison is made, and 19 is located as the 14th term in the original list. ■

We now specify the steps of the binary search algorithm. To search for the integer x in the list $a_1, a_2, \ldots, a_n$, where $a_1 < a_2 < \cdots < a_n$, begin by comparing x with the middle term of the sequence, a_m, where $m = \lfloor (n+1)/2 \rfloor$. (Recall that $\lfloor x \rfloor$ is the greatest integer not exceeding x.) If $x > a_m$, the search for x can be restricted to the second half of the sequence, which is $a_{m+1}, a_{m+2}, \ldots, a_n$. If x is not greater than a_m, the search for x can be restricted to the first half of the sequence, which is $a_1, a_2, \ldots, a_m$.

The search has now been restricted to a list with no more than $\lceil n/2 \rceil$ elements. Using the same procedure, compare x to the middle term of the restricted list. Then restrict the search to the first or second half of the list. Repeat this process until a list with one term is obtained. Then determine whether this term is x. Pseudocode for the binary search algorithm is displayed as Algorithm 3.

ALGORITHM 3 The Binary Search Algorithm.

```
procedure binary search($x$: integer, $a_1, a_2, \ldots, a_n$: increasing integers)
$i := 1$  {$i$ is left endpoint of search interval}
$j := n$  {$j$ is right endpoint of search interval}
while $i < j$
begin
    $m := \lfloor (i + j)/2 \rfloor$
    if $x > a_m$ then $i := m + 1$
    else $j := m$
end
if $x = a_i$ then location := $i$
else location := 0
{location is the subscript of term equal to $x$, or 0 if $x$ is not found}
```

Algorithm 3 proceeds by successively narrowing down the part of the sequence being searched. At any given stage only the terms beginning with a_i and ending with a_j are under consideration. In other words, i and j are the smallest and largest subscripts of the remaining terms, respectively. Algorithm 3 continues narrowing the part of the sequence being searched until only one term of the sequence remains. When this is done, a comparison is made to see whether this term equals x.

Exercises

1. List all the steps used by Algorithm 1 to find the maximum of the list 1, 8, 12, 9, 11, 2, 14, 5, 10, 4.
2. Determine which characteristics of an algorithm the following procedures have and which they lack.
   a)
```
procedure double($n$: positive integer)
while $n > 0$
    $n := 2n$
```
   b)
```
procedure divide($n$: positive integer)
while $n \geq 0$
begin
    $m := 1/n$
    $n := n - 1$
end
```

c)
```
procedure sum(n: positive integer)
sum := 0
while i < 10
  sum := sum + i
```

d)
```
procedure choose(a,b: integers)
x := either a or b
```

3. Devise an algorithm that finds the sum of all the integers in a list.

4. Devise an algorithm to compute x^n, where x is a real number and n is an integer. (*Hint:* First give a procedure for computing x^n when n is nonnegative by successive multiplication by x, starting with 1. Then extend this procedure, and use the fact that $x^{-n} = 1/x^n$ to compute x^n when n is negative.)

5. Describe an algorithm that interchanges the values of the variables x and y, using only assignments. What is the minimum number of assignment statements needed to do this?

6. Describe an algorithm that uses only assignment statements that replaces the triple (x, y, z) with (y, z, x). What is the minimum number of assignment statements needed?

7. List all the steps used to search for 9 in the sequence 1, 3, 4, 5, 6, 8, 9, 11 using

a) a linear search. **b)** a binary search.

8. List all the steps used to search for 7 in the sequence given in Exercise 7.

9. Describe an algorithm that inserts an integer x in the appropriate position into the list $a_1, a_2, \ldots, a_n$ of integers that are in increasing order.

10. Describe an algorithm for finding the smallest integer in a finite sequence of natural numbers.

11. Describe an algorithm that locates the first occurrence of the largest element in a finite list of integers, where the integers in the list are not necessarily distinct.

12. Describe an algorithm that locates the last occurrence of the smallest element in a finite list of integers, where the integers in the list are not necessarily distinct.

13. Describe an algorithm that produces the maximum, median, mean, and minimum of a set of three integers. (The **median** of a set of integers is the middle element in the list when these integers are listed in order of increasing size. The **mean** of a set of integers is the sum of the integers divided by the number of integers in the set.)

14. Describe an algorithm for finding both the largest and the smallest integers in a finite sequence of integers.

15. Describe an algorithm that puts the first three terms of a sequence of integers of arbitrary length in increasing order.

16. Describe an algorithm to find the longest word in an English sentence (where a word is a string of letters and a sentence is a list of words, separated by blanks).

17. Describe an algorithm that determines whether a function from a finite set to another finite set is onto.

18. Describe an algorithm that determines whether a function from a finite set to another finite set is one-to-one.

19. Describe an algorithm that will count the number of 1s in a bit string by examining each bit of the string to determine whether it is a 1 bit.

20. Change Algorithm 3 so that the binary search procedure compares x to a_m at each stage of the algorithm, with the algorithm terminating if $x = a_m$. What advantage does this version of the algorithm have?

21. The **ternary search algorithm** locates an element in a list of increasing integers by successively splitting the list into three sublists of equal (or as close to equal as possible) size, and restricting the search to the appropriate piece. Specify the steps of this algorithm.

22. Specify the steps of an algorithm that locates an element in a list of increasing integers by successively splitting the list into four sublists of equal (or as close to equal as possible) size, and restricting the search to the appropriate piece.

23. A **mode** of a list of integers is an element that occurs at least as often as each of the other elements. Devise an algorithm that finds a mode in a list of nondecreasing integers.

24. Devise an algorithm that finds all modes (defined in Exercise 23) in a list of nondecreasing integers.

25. Devise an algorithm that finds the first term of a sequence of integers that equals some previous term in the sequence.

26. Devise an algorithm that finds all terms of a finite sequence of integers that are greater than the sum of all previous terms of the sequence.

27. Devise an algorithm that finds the first term of a sequence of positive integers that is less than the immediately preceding term of the sequence.

2.2

Complexity of Algorithms

INTRODUCTION

When does an algorithm provide a satisfactory solution to a problem? First, it must always produce the correct answer. How this can be demonstrated will be discussed in

Chapter 3. Second, it should be efficient. The efficiency of algorithms will be discussed in this section.

How can the efficiency of an algorithm be analyzed? One measure of efficiency is the time used by a computer to solve a problem using the algorithm, when input values are of a specified size. A second measure is the amount of computer memory required to implement the algorithm when input values are of a specified size.

Questions such as these involve the **computational complexity** of the algorithm. An analysis of the time required to solve a problem of a particular size involves the **time complexity** of the algorithm. An analysis of the computer memory required involves the **space complexity** of the algorithm. Considerations of the time and space complexity of an algorithm are essential when algorithms are implemented. It is obviously important to know whether an algorithm will produce an answer in a microsecond, a minute, or a billion years. Likewise, the required memory must be available to solve a problem, so that space complexity must be taken into account.

Considerations of space complexity are tied in with the particular data structures used to implement the algorithm. Because data structures are not dealt with in detail in this book, space complexity will not be considered. We will restrict our attention to time complexity.

The time complexity of an algorithm can be expressed in terms of the number of operations used by the algorithm when the input has a particular size. The operations used to measure time complexity can be the comparison of integers, the addition of integers, the multiplication of integers, the division of integers, or any other basic operation.

Time complexity is described in terms of the number of operations required instead of actual computer time because of the difference in time needed for different computers to perform basic operations. Moreover, it is quite complicated to break all operations down to the basic bit operations that a computer uses. Furthermore, the fastest computers in existence can perform basic bit operations (for instance, adding, multiplying, comparing, or exchanging two bits) in 10^{-9} second (1 nanosecond), but personal computers may require 10^{-6} second (1 microsecond), which is 1000 times as long, to do the same operations.

We illustrate how to analyze the time complexity of an algorithm by considering Algorithm 1 of Section 2.1, which finds the maximum of a finite set of integers.

EXAMPLE 1 Describe the time complexity of Algorithm 1 of Section 2.1 for finding the maximum element in a set.

Solution: The number of comparisons will be used as the measure of the time complexity of the algorithm, since comparisons are the basic operations used.

To find the maximum element of a set with n elements, listed in an arbitrary order, the temporary maximum is first set equal to the initial term in the list. Then, after a comparison has been done to determine that the end of the list has not yet been reached, the temporary maximum and second term are compared, updating the temporary maximum to the value of the second term if it is larger. This procedure is continued, using two additional comparisons for each term of the list—one to determine that the end of the list has not been reached and another to determine whether to update the temporary maximum. Since two comparisons are used for each of the second through the nth elements and one more comparison is used to exit the loop when $i = n + 1$, exactly $2(n-1)+1 = 2n-1$ comparisons are used whenever this algorithm is applied. Hence, the algorithm for finding the maximum of a set of n elements has time complexity $O(n)$, measured in terms of the number of comparisons used. ■

Next, the time complexity of searching algorithms will be analyzed.

EXAMPLE 2 Describe the time complexity of the linear search algorithm.

Solution: The number of comparisons used by the algorithm will be taken as the measure of the time complexity. At each step of the loop in the algorithm, two comparisons are performed—one to see whether the end of the list has been reached and one to compare the element x with a term of the list. Finally, one more comparison is made outside the loop. Consequently, if $x = a_i$, $2i + 1$ comparisons are used. The most comparisons, $2n + 2$, are required when the element is not in the list. In this case, $2n$ comparisons are used to determine that x is not a_i, for $i = 1, 2, \ldots, n$, an additional comparison is used to exit the loop, and one comparison is made outside the loop. So when x is not in the list, a total of $2n + 2$ comparisons are used. Hence, a linear search requires at most $O(n)$ comparisons. ■

The type of complexity analysis done in Example 2 is a **worst-case** analysis. By the worst-case performance of an algorithm, we mean the largest number of operations needed to solve the given problem using this algorithm on input of specified size. Worst-case analysis tells us how many operations an algorithm requires to guarantee that it will produce a solution.

EXAMPLE 3 Describe the time complexity of the binary search algorithm.

Solution: For simplicity, assume there are $n = 2^k$ elements in the list $a_1, a_2, \ldots, a_n$, where k is a nonnegative integer. Note that $k = \log n$. (If n, the number of elements in the list, is not a power of 2, the list can be considered part of a larger list with 2^{k+1} elements, where $2^k < n < 2^{k+1}$. Here 2^{k+1} is the smallest power of 2 larger than n.)

At each stage of the algorithm, i and j, the locations of the first term and the last term of the restricted list at that stage, are compared to see whether the restricted list has more than one term. If $i < j$, a comparison is done to determine whether x is greater than the middle term of the restricted list.

At the first stage the search is restricted to a list with 2^{k-1} terms. So far, two comparisons have been used. This procedure is continued, using two comparisons at each stage to restrict the search to a list with half as many terms. In other words, two comparisons are used at the first stage of the algorithm when the list has 2^k elements, two more when the search has been reduced to a list with 2^{k-1} elements, two more when the search has been reduced to a list with 2^{k-2} elements, and so on, until two comparisons are used when the search has been reduced to a list with $2^1 = 2$ elements. Finally, when one term is left in the list, one comparison tells us that there are no additional terms left, and one more comparison is used to determine if this term is x.

Hence, at most $2k + 2 = 2 \log n + 2$ comparisons are required to perform a binary search when the list being searched has 2^k elements. (If n is not a power of 2, the original list is expanded to a list with 2^{k+1} terms, where $k = \lfloor \log n \rfloor$, and the search requires at most $2\lceil \log n \rceil + 2$ comparisons.) Consequently, a binary search requires at most $O(\log n)$ comparisons. From this analysis it follows that the binary search algorithm is more efficient, in the worst case, than a linear search. ■

Another important type of complexity analysis, besides worst-case analysis, is called **average-case** analysis. The average number of operations used to solve the problem over all inputs of a given size is found in this type of analysis. Average-case time complexity analysis is usually much more complicated than worst-case analysis. However, the average-case analysis for the linear search algorithm can be done without difficulty, as shown in Example 4.

EXAMPLE 4 Describe the average-case performance of the linear search algorithm, assuming that the element x is in the list.

Solution: There are n types of possible inputs when x is known to be in the list. If x is the first term of the list, three comparisons are needed, one to determine whether the end of the list has been reached, one to compare x and the first term, and one outside the loop. If x is the second term of the list, two more comparisons are needed, so that a total of five comparisons are used. In general, if x is the ith term of the list, two comparisons will be used at each of the i steps of the loop, and one outside the loop, so that a total of $2i + 1$ comparisons are needed. Hence, the average number of comparisons used equals

$$\frac{3 + 5 + 7 + \cdots + (2n + 1)}{n} = \frac{2(1 + 2 + 3 + \cdots + n) + n}{n}.$$

In Section 3.2 we will show that

$$1 + 2 + 3 + \cdots + n = \frac{n(n + 1)}{2}.$$

Hence, the average number of comparisons used by the linear search algorithm (when x is known to be in the list) is

$$\frac{2[n(n + 1)/2]}{n} + 1 = n + 2,$$

which is $O(n)$.

Remark: In this analysis it has been assumed that x is in the list being searched and it is equally likely that x is in any position. It is also possible to do an average-case analysis of this algorithm when x may not be in the list (see Exercise 13 at the end of this section). ■

Table 1 displays some common terminology used to describe the time complexity of algorithms. For instance, an algorithm is said to have **exponential complexity** if it has time complexity $O(b^n)$, where $b > 1$, measured in terms of some specified type of operation. Similarly, an algorithm with time complexity $O(n^b)$ is said to have **polynomial complexity.** The linear search algorithm has **linear** (worst- or average-case)

TABLE 1 Commonly Used Terminology for the Complexity of Algorithms.

Complexity	*Terminology*
$O(1)$	Constant complexity
$O(\log n)$	Logarithmic complexity
$O(n)$	Linear complexity
$O(n \log n)$	$n \log n$ complexity
$O(n^b)$	Polynomial complexity
$O(b^n)$, where $b > 1$	Exponential complexity
$O(n!)$	Factorial complexity

complexity and the binary search algorithm has **logarithmic** (worst-case) **complexity**, measured in terms of the number of comparisons used.

A problem that is solvable using an algorithm with polynomial worst-case complexity is called **tractable,** since the expectation is that the algorithm will produce the solution to the problem for reasonably sized input in a relatively short time. However, if the polynomial in the big-O estimate has high degree (such as degree 100) or if the coefficients are extremely large, the algorithm may take an extremely long time to solve the problem. Consequently, that a problem can be solved using an algorithm with polynomial worst-case time complexity is no guarantee that the problem can be solved in a reasonable amount of time for even relatively small input values. Fortunately, in practice, the degree and coefficients of polynomials in such estimates are small.

The situation is much worse for problems that cannot be solved using an algorithm with worst-case polynomial time complexity. Such problems are called **intractable.** Usually, but not always, an extremely large amount of time is required to solve the problem for the worst cases of even small input values. In practice, however, there are situations where an algorithm with worst-case time complexity may be able to solve a problem much more quickly for most cases than for its worst case. When we are willing to allow that some, perhaps small, number of cases may not be solved in a reasonable amount of time, the average-case time complexity is a better measure of how long an algorithm takes to solve a problem. Many problems important in industry are thought to be intractable but can be practically solved for essentially all sets of input that arise in daily life. Another way that intractable problems are handled when they arise in practical applications is that instead of looking for exact solutions of a problem, approximate solutions are sought. It may be the case that fast algorithms exist for finding such approximate solutions, perhaps even with a guarantee that they do not differ by very much from an exact solution.

Some problems even exist for which it can be shown that no algorithm exists for solving them. Such problems are called **unsolvable** (as opposed to **solvable** problems that can be solved using an algorithm). The first proof that there are unsolvable problems was provided by the great English mathematician and computer scientist Alan Turing. The problem he showed unsolvable is the **halting problem.** This problem takes as its input a program together with input to this program. The problem asks whether the program will halt when executed with the input to the program. We will study the halting problem in Section 3.1. (A biography of Alan Turing and a description of some of his other work can be found in Chapter 10.)

The study of the complexity of algorithms goes far beyond what we can describe here. Note, however, that many solvable problems are believed to have the property that no algorithm with polynomial worst-case time complexity solves them, but that once a solution is known, it can be checked in polynomial time. Problems for which a solution can be checked in polynomial time are said to belong to the **class NP** (tractable problems are said to belong to **class P**). There is also an important class of problems, called **NP-complete problems**, with the property that if any of these problems can be solved by a polynomial worst-case time algorithm, then all can be solved by polynomial worst-case time algorithms. Despite extensive research, no polynomial worst-case time algorithm has been found for any problem in this class. It is generally accepted, although not proven, that no NP-complete problem can be solved in polynomial time. For more information about the complexity of algorithms, consult the references, including [CoLeRi 90], for this section listed at the end of this book.

Note that a big-O estimate of the time complexity of an algorithm expresses how the time required to solve the problem changes as the input grows in size. In practice, the best estimate (that is, with the smallest reference function) that can be shown is used. However, big-O estimates of time complexity cannot be directly translated into the actual amount of computer time used. One reason is that a big-O estimate $f(n)$ is $O(g(n))$, where $f(n)$ is the time complexity of an algorithm and $g(n)$ is a reference function, means that $f(n) \leq Cg(n)$ when $n > k$, where C and k are constants. So without knowing the constants C and k in the inequality, this estimate cannot be used to determine an upper bound on the number of operations used. Moreover, as remarked before, the time required for an operation depends on the type of operation and the computer being used. (Also note that a big-O estimate on the time complexity of an algorithm provides an upper, but not a lower, bound, on the worst-case time required for the algorithm as a function of the input size. To provide a lower bound, a big-Theta estimate should be used. However, for simplicity, we will use big-O estimates when describing the time complexity of algorithms, with the understanding that big-Theta estimates would provide more information.)

However, the time required for an algorithm to solve a problem of a specified size can be determined if all operations can be reduced to the bit operations used by the computer. Table 2 displays the time needed to solve problems of various sizes with

TABLE 2 **The Computer Time Used by Algorithms.**

Problem Size	*Bit Operations Used*					
n	$\log n$	n	$n \log n$	n^2	2^n	$n!$
10	3×10^{-9} s	10^{-8} s	3×10^{-8} s	10^{-7} s	10^{-6} s	3×10^{-3} s
10^2	7×10^{-9} s	10^{-7} s	7×10^{-7} s	10^{-5} s	4×10^{13} yr	*
10^3	1.0×10^{-8} s	10^{-6} s	1×10^{-5} s	10^{-3} s	*	*
10^4	1.3×10^{-8} s	10^{-5} s	1×10^{-4} s	10^{-1} s	*	*
10^5	1.7×10^{-8} s	10^{-4} s	2×10^{-3} s	10 s	*	*
10^6	2×10^{-8} s	10^{-3} s	2×10^{-2} s	17 min	*	*

an algorithm using the indicated number of bit operations. Times of more than 10^{100} years are indicated with an asterisk. (In Section 2.4 the number of bit operations used to add and multiply two integers will be discussed.) In the construction of this table, each bit operation is assumed to take 10^{-9} second, which is the time required for a bit operation using the fastest computers today. In the future, these times will decrease as faster computers are developed.

It is important to know how long a computer will need to solve a problem. For instance, if an algorithm requires 10 hours, it may be worthwhile to spend the computer time (and money) required to solve this problem. But, if an algorithm requires 10 billion years to solve a problem, it would be unreasonable to use resources to implement this algorithm. One of the most interesting phenomena of modern technology is the tremendous increase in the speed and memory space of computers. Another important factor that decreases the time needed to solve problems on computers is **parallel processing,** which is the technique of performing sequences of operations simultaneously. Because of the increased speed of computation, increases in computer memory, and the use of algorithms that take advantage of parallel processing, problems that were considered impossible to solve 5 years ago are now routinely solved, and certainly 5 years from now this statement will still be true.

Exercises

1. How many comparisons are used by the algorithm given in Exercise 10 of Section 2.1 to find the smallest natural number in a sequence of n natural numbers?

2. Write the algorithm that puts the first four terms of a list of arbitrary length in increasing order. Show that this algorithm has time complexity $O(1)$ in terms of the number of comparisons used.

3. Suppose that an element is known to be among the first four elements in a list of 32 elements. Would a linear search or a binary search locate this element more rapidly?

4. Determine the number of multiplications used to find x^{2^k} starting with x and successively squaring (to find x^2, x^4, and so on). Is this a more efficient way to find x^{2^k} than by multiplying x by itself the appropriate number of times?

5. Give a big-O estimate for the number of comparisons used by the algorithm that determines the number of 1s in a bit string by examining each bit of the string to determine whether it is a 1 bit (see Exercise 19 of Section 2.1).

***6.** **a)** Show that the following algorithm determines the number of 1 bits in the bit string S.

```
procedure bitcount(S: bit string)
count := 0
while $S \neq 0$
begin
   count := count + 1
   $S := S \wedge (S - 1)$
end {count is the number of 1s in S}
```

Here $S - 1$ is the bit string obtained by changing the rightmost 1 bit of S to a 0 and all the 0 bits to the right of this to 1s. [Recall that $S \wedge (S - 1)$ is the bitwise *AND* of S and $S - 1$.]

b) How many bitwise *AND* operations are needed to find the number of 1 bits in a string S?

7. The conventional algorithm for evaluating a polynomial $a_nx^n + a_{n-1}x^{n-1} + \cdots + a_1x + a_0$ at $x = c$ can be expressed in pseudocode by

```
procedure polynomial($c, a_0, a_1, \ldots, a_n$: real
   numbers)
power := 1
$y := a_0$
for $i := 1$ to $n$
begin
   power := power * c
   $y := y + a_i$ * power
end {$y = a_nc^n + a_{n-1}c^{n-1} + \cdots + a_1c + a_0$}
```

where the final value of y is the value of the polynomial at $x = c$.

a) Evaluate $3x^2 + x + 1$ at $x = 2$ by working through each step of the algorithm.

b) Exactly how many multiplications and additions are used to evaluate a polynomial of degree n at $x = c$? (Do not count additions used to increment the loop variable.)

8. There is a more efficient algorithm (in terms of the number of multiplications and additions used) for evaluating polynomials than the conventional algorithm described in the previous exercise. It is called **Horner's method.** The following pseudocode shows how to use this method to find the value of $a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$ at $x = c$.

```
procedure Horner($c, a_0, a_1, a_2, \ldots, a_n$: real
    numbers)
$y := a_n$
for $i := 1$ to $n$
    $y := y * c + a_{n-i}$
{$y = a_n c^n + a_{n-1} c^{n-1} + \cdots + a_1 c + a_0$}
```

a) Evaluate $3x^2 + x + 1$ at $x = 2$ by working through each step of the algorithm.

b) Exactly how many multiplications and additions are used by this algorithm to evaluate a polynomial of degree n at $x = c$? (Do not count additions used to increment the loop variable.)

9. How large a problem can be solved in 1 second using an algorithm that requires $f(n)$ bit operations, where each bit operation is carried out in 10^{-9} second, with the following values for $f(n)$?

a) $\log n$ b) n c) $n \log n$

d) n^2 e) 2^n f) $n!$

10. How much time does an algorithm take to solve a problem of size n if this algorithm uses $2n^2 + 2^n$ bit operations, each requiring 10^{-9} second, with the following values of n?

a) 10 b) 20

c) 50 d) 100

11. How much time does an algorithm using 2^{50} bit operations need if each bit operation takes the following amount of time?

a) 10^{-6} second

b) 10^{-9} second

c) 10^{-12} second

12. Determine the least number of comparisons, or best-case performance,

a) required to find the maximum of a sequence of n integers, using Algorithm 1 of Section 2.1.

b) used to locate an element in a list of n terms with a linear search.

c) used to locate an element in a list of n terms using a binary search.

13. Analyze the average-case performance of the linear search algorithm, if exactly half the time element x is not in the list and if x is in the list it is equally likely to be in any position.

14. An algorithm is called **optimal** for the solution of a problem with respect to a specified operation if there is no algorithm for solving this problem using fewer operations.

a) Show that Algorithm 1 in Section 2.1 is an optimal algorithm with respect to the number of comparisons of integers. (*Note:* Comparisons used for bookkeeping in the loop are not of concern here.)

b) Is the linear search algorithm optimal with respect to the number of comparisons of integers (not including comparisons used for bookkeeping in the loop)?

15. Describe the worst-case time complexity, measured in terms of comparisons, of the ternary search algorithm described in Exercise 21 of Section 2.1.

16. Describe the worst-case time complexity, measured in terms of comparisons, of the search algorithm described in Exercise 22 of Section 2.1.

17. Analyze the worst-case time complexity of the algorithm you devised in Exercise 23 of Section 2.1 for locating a mode in a list of nondecreasing integers.

18. Analyze the worst-case time complexity of the algorithm you devised in Exercise 24 of Section 2.1 for locating all modes in a list of nondecreasing integers.

19. Analyze the worst-case time complexity of the algorithm you devised in Exercise 25 of Section 2.1 for finding the first term of a sequence of integers equal to some previous term.

20. Analyze the worst-case time complexity of the algorithm you devised in Exercise 26 of Section 2.1 for finding all terms of a sequence that are greater than the sum of all previous terms.

21. Analyze the worst-case time complexity of the algorithm you devised in Exercise 27 of Section 2.1 for finding the first term of a sequence less than the immediately preceding term.

2.3

The Integers and Division

INTRODUCTION

The part of discrete mathematics involving the integers and their properties belongs to the branch of mathematics called **number theory.** This section is the beginning of a three-section introduction to number theory. In this section we will review some basic concepts of number theory, including divisibility, greatest common divisors, and modular arithmetic. In Section 2.4 we will describe several important algorithms from number theory, tying together the material in Sections 2.1 and 2.2 on algorithms and their complexity with the notions introduced in this section. For example, we will introduce algorithms for finding the greatest common divisor of two positive integers and for performing computer arithmetic using binary expansions. Finally, in Section 2.5, we will continue our study of number theory by introducing some important results and their applications to computer arithmetic and cryptology, the study of secret messages.

The ideas that we will develop in this section are based on the notion of divisibility. One important concept based on divisibility is that of a prime number. A prime is an integer greater than 1 that is divisible only by 1 and by itself. Determining whether an integer is prime is important in applications to cryptology. An important theorem from number theory, the Fundamental Theorem of Arithmetic, asserts that every positive integer can be written uniquely as the product of prime numbers. Factoring integers into their prime factors is important in cryptology. Division of an integer by a positive integer produces a quotient and a remainder. Working with these remainders leads to modular arithmetic, which is used throughout computer science. We will discuss three applications of modular arithmetic in this section: generating pseudorandom numbers, assigning computer memory locations to files, and encrypting and decrypting messages.

DIVISION

When one integer is divided by a second, nonzero integer, the quotient may or may not be an integer. For example, $12/3 = 4$ is an integer, whereas $11/4 = 2.75$ is not. This leads to the following definition.

DEFINITION 1. **If a and b are integers with $a \neq 0$, we say that a *divides* b if there is an integer c such that $b = ac$. When a divides b we say that a is a *factor* of b and that b is a *multiple* of a. The notation $a \mid b$ denotes that a divides b. We write $a \nmid b$ when a does not divide b.**

In Figure 1 a number line indicates which integers are divisible by the positive integer d.

EXAMPLE 1 Determine whether $3 \nmid 7$ and whether $3 \mid 12$.

Solution: It follows that $3 \nmid 7$, since $7/3$ is not an integer. On the other hand, $3 \mid 12$ since $12/3 = 4$. ■

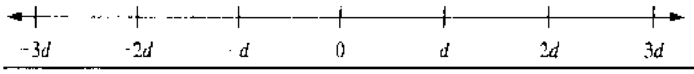


FIGURE 1 **Integers Divisible by the Positive Integer *d*.**

EXAMPLE 2 Let n and d be positive integers. How many positive integers not exceeding n are divisible by d?

Solution: The positive integers divisible by d are all the integers of the form dk, where k is a positive integer. Hence, the number of positive integers divisible by d that do not exceed n equals the number of integers k with $0 < dk \leq n$, or with $0 < k \leq n/d$. Therefore, there are $\lfloor n/d \rfloor$ positive integers not exceeding n that are divisible by d. ■

Some of the basic properties of divisibility of integers are given in Theorem 1.

THEOREM 1 **Let a, b, and c be integers. Then**

1. **if $a \mid b$ and $a \mid c$, then $a \mid (b + c)$;**
2. **if $a \mid b$, then $a \mid bc$ for all integers c;**
3. **if $a \mid b$ and $b \mid c$, then $a \mid c$.**

Proof: Suppose that $a \mid b$ and $a \mid c$. Then, from the definition of divisibility, it follows that there are integers s and t with $b = as$ and $c = at$. Hence,

$$b + c = as + at = a(s + t).$$

Therefore, a divides $b + c$. This establishes part (1) of the theorem. The proofs of parts (2) and (3) are left as exercises for the reader. □

PRIMES

Every positive integer greater than 1 is divisible by at least two integers, since a positive integer is divisible by 1 and by itself. Integers that have exactly two different positive integer factors are called **primes.**

DEFINITION 2. A positive integer p greater than 1 is called *prime* if the only positive factors of p are 1 and p. A positive integer that is greater than 1 and is not prime is called *composite*.

EXAMPLE 3 The integer 7 is prime since its only positive factors are 1 and 7, whereas the integer 9 is composite since it is divisible by 3. ■

The primes less than 100 are 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, and 97.

The primes are the building blocks of positive integers, as the Fundamental Theorem of Arithmetic shows. The proof will be given in Section 3.2.

THEOREM 2 **THE FUNDAMENTAL THEOREM OF ARITHMETIC Every positive integer can be written uniquely as the product of primes, where the prime factors are written in order of increasing size. (Here, a product can have zero, one, or more than one prime factor.)**

The next example gives some prime factorizations of integers.

EXAMPLE 4 The prime factorizations of 100, 641, 999, and 1024 are given by

$$\begin{aligned} 100 &= 2 \cdot 2 \cdot 5 \cdot 5 = 2^2 5^2, \\ 641 &= 641, \\ 999 &= 3 \cdot 3 \cdot 3 \cdot 37 = 3^3 \cdot 37, \\ 1024 &= 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 2^{10}. \end{aligned}$$

■

It is often important to show that a given integer is prime. For instance, in cryptology large primes are used in some methods for making messages secret. One procedure for showing that an integer is prime is based on the following observation.

THEOREM 3 **If n is a composite integer, then n has a prime divisor less than or equal to $\sqrt{n}$.**

Proof: If n is composite, it has a factor a with $1 < a < n$. Hence, $n = ab$, where both a and b are positive integers greater than 1. We see that $a \le \sqrt{n}$ or $b \le \sqrt{n}$, since otherwise $ab > \sqrt{n} \cdot \sqrt{n} = n$. Hence, n has a positive divisor not exceeding $\sqrt{n}$. This divisor is either prime or, by the Fundamental Theorem of Arithmetic, has a prime divisor. In either case, n has a prime divisor less than or equal to $\sqrt{n}$. □

From Theorem 3, it follows that an integer is prime if it is not divisible by any prime less than or equal to its square root. In the following example this observation is used to show that 101 is prime.

EXAMPLE 5 Show that 101 is prime.

Solution: The only primes not exceeding $\sqrt{101}$ are 2, 3, 5, and 7. Since 101 is not divisible by 2, 3, 5, or 7 (the quotient of 101 and each of these integers is not an integer), it follows that 101 is prime. ■

Since every integer has a prime factorization, it would be useful to have a procedure for finding this prime factorization. Consider the problem of finding the prime factorization of n. Begin by dividing n by successive primes, starting with the smallest prime, 2. If n has a prime factor, then by Theorem 3 a prime factor p not exceeding $\sqrt{n}$ will be found. So, if no prime factor not exceeding $\sqrt{n}$ is found, then n is prime. Otherwise, if a prime factor p is found, continue by factoring n/p. Note that n/p has no prime factors less than p. Again, if n/p has no prime factor greater than or equal to p and not exceeding its square root, then it is prime. Otherwise, if it has a prime factor q, continue by factoring $n/(pq)$. This procedure is continued until the factorization has been reduced to a prime. This procedure is illustrated in the following example.

EXAMPLE 6 Find the prime factorization of 7007.

Solution: To find the prime factorization of 7007, first perform divisions of 7007 by successive primes, beginning with 2. None of the primes 2, 3, and 5 divides 7007. However, 7 divides 7007, with $7007/7 = 1001$. Next, divide 1001 by successive primes, beginning with 7. It is immediately seen that 7 also divides 1001, since $1001/7 = 143$. Continue by dividing 143 by successive primes, beginning with 7. Although 7 does not divide 143, 11 does divide 143, and $143/11 = 13$. Since 13 is prime, the procedure is completed. It follows that the prime factorization of 7007 is $7 \cdot 7 \cdot 11 \cdot 13 = 7^2 \cdot 11 \cdot 13$. ■

web

Prime numbers were studied in ancient times for philosophical reasons. Today, there are now highly practical reasons for their study. In particular, large primes play a crucial role in cryptography, as we will see in Section 2.5. It has long been known that there are infinitely many primes, a fact we will prove in Section 3.1. Since there are infinitely many primes, given any positive integer there are primes greater than this integer. There is an ongoing quest to discover larger and larger prime numbers; for almost all the last 300 years, the largest prime known has been an integer of the special form $2^p - 1$, where p is also prime. Such primes are called **Mersenne primes**, after the French monk Marin Mersenne, who studied them in the seventeenth century. The reason that the largest known prime has usually been a Mersenne prime is that there is an extremely efficient test, known as the Lucas–Lehmer test, for determining whether $2^p - 1$ is prime. Furthermore, it is not currently possible to test numbers not of certain special forms anywhere near as quickly to determine whether they are prime.

EXAMPLE 7 The numbers $2^2 - 1 = 3$, $2^3 - 1 = 7$, and $2^5 - 1 = 31$ are Mersenne primes, while $2^{11} - 1 = 2047$ is not a Mersenne prime since $2047 = 23 \cdot 89$. ■

Progress in finding Mersenne primes has been steady since computers were invented. As of late 1998, 37 different Mersenne primes were known, with six found

---

web

Marin Mersenne (1588–1648). Mersenne was born in Maine, France, into a family of laborers and attended the College of Mans and the Jesuit College at La Flèche. He continued his education at the Sorbonne, studying theology from 1609 to 1611. He joined the religious order of the Minims in 1611, a group whose name comes from the word *minimi* (the members of this group considered themselves the least religious order). Besides prayer, the members of this group devoted their energy to scholarship and study. In 1612 he became a priest at the Place Royale in Paris; between 1614 and 1618 he taught philosophy at the Minim Convent at Nevers. He returned to Paris in 1619, where his cell in the Minims de l'Annociade became a place for meetings of French scientists, philosophers, and mathematicians, including Fermat and Pascal. Mersenne corresponded extensively with scholars throughout Europe, serving as a clearinghouse for mathematical and scientific knowledge, a function later served by mathematical journals (and today also by the Internet). Mersenne wrote books covering mechanics, mathematical physics, mathematics, music, and acoustics. He studied prime numbers and tried unsuccessfully to construct a formula representing all primes. In 1644 Mersenne claimed that $2^p - 1$ is prime for $p = 2, 3, 5, 7, 13, 17, 19, 31, 67, 127, 257$ but is composite for all other primes less than 257. It took over 300 years to determine that Mersenne's claim was wrong five times. Specifically, $2^p - 1$ is not prime for $p = 67$ and $p = 257$ but is prime for $p = 61$, $p = 87$, and $p = 107$. It is also noteworthy that Mersenne defended two of the most famous men of his time, Descartes and Galileo, from religious critics. He also helped expose alchemists and astrologers as frauds.

since 1990. The largest Mersenne prime known (as of late 1998) is $2^{3021377} - 1$, a number with 909,526 digits. A communal effort, the Great Internet Mersenne Prime Search (GIMPS), has been organized to look for new Mersenne primes. By the way, even the search for Mersenne primes has practical implications. One quality control test for supercomputers has been to replicate the Lucas–Lehmer test that establishes the primality of a large Mersenne prime.

Using trial division with Theorem 3 gives procedures for factoring and for primality testing. However, these procedures are not efficient algorithms; many much more practical and efficient algorithms for these tasks have been developed. Factoring and primality testing have become important in the applications of number theory to cryptography. This has led to a great interest in developing efficient algorithms for both tasks. Clever procedures have been devised in the last 25 years for efficiently generating large primes. However, even though powerful new factorization methods have been developed in the same time frame, factoring large numbers remains extraordinarily more time consuming. Nevertheless, the challenge of factoring large numbers interests many people. There is a communal effort on the Internet to factor large numbers, especially those of the special form $k^n \pm 1$, where k is a small positive integer and n is a large positive integer (such numbers are called Cunningham numbers). At any given time, there is a list of the "Ten Most Wanted" large numbers of this type awaiting factorization.

THE DIVISION ALGORITHM

We have seen that an integer may or may not be divisible by another. However, when an integer is divided by a positive integer, there always is a quotient and a remainder, as the division algorithm shows.

THEOREM 4 THE DIVISION ALGORITHM **Let a be an integer and d a positive integer. Then there are unique integers q and r, with $0 \leq r < d$, such that $a = dq + r$.**

Remark: Theorem 4 is not really an algorithm. (Why not?) Nevertheless, we use its traditional name.

DEFINITION 3. In the equality given in the division algorithm, d is called the *divisor*, a is called the *dividend*, q is called the *quotient*, and r is called the *remainder*.

The following two examples illustrate the division algorithm.

EXAMPLE 8 What are the quotient and remainder when 101 is divided by 11?

Solution: We have

$$101 = 11 \cdot 9 + 2.$$

Hence, the quotient when 101 is divided by 11 is 9, and the remainder is 2. ■

EXAMPLE 9 What are the quotient and remainder when -11 is divided by 3?

Solution: We have

$$-11 = 3(-4) + 1.$$

Hence, the quotient when -11 is divided by 3 is -4, and the remainder is 1.

Note that the remainder cannot be negative. Consequently, the remainder is *not* -2, even though

$$-11 = 3(-3) - 2,$$

since $r = -2$ does not satisfy $0 \leq r < 3$. ■

Note that the integer a is divisible by the integer d if and only if the remainder is zero when a is divided by d.

GREATEST COMMON DIVISORS AND LEAST COMMON MULTIPLES

The largest integer that divides both of two integers is called the **greatest common divisor** of these integers.

DEFINITION 4. Let a and b be integers, not both zero. The largest integer d such that $d \mid a$ and $d \mid b$ is called the *greatest common divisor* of a and b. The greatest common divisor of a and b is denoted by $\gcd(a, b)$.

The greatest common divisor of two integers, not both zero, exists because the set of common divisors of these integers is finite. One way to find the greatest common divisor of two integers is to find all the positive common divisors of both integers and then take the largest divisor. This is done in the following examples. Later, a more efficient method of finding greatest common divisors will be given.

EXAMPLE 10 What is the greatest common divisor of 24 and 36?

Solution: The positive common divisors of 24 and 36 are 1, 2, 3, 4, 6, and 12. Hence, $\gcd(24, 36) = 12$. ■

EXAMPLE 11 What is the greatest common divisor of 17 and 22?

Solution: The integers 17 and 22 have no positive common divisors other than 1, so that $\gcd(17, 22) = 1$. ■

Since it is often important to specify that two integers have no common positive divisor other than 1, we have the following definition.

DEFINITION 5. The integers a and b are ***relatively prime*** if their greatest common divisor is 1.

EXAMPLE 12 From Example 11 it follows that the integers 17 and 22 are relatively prime, since $\gcd(17, 22) = 1$. ■

Since we often need to specify that no two integers in a set of integers have a common positive divisor greater than 1, we make the following definition.

DEFINITION 6. The integers $a_1, a_2, \ldots, a_n$ are ***pairwise relatively prime*** if $\gcd(a_i, a_j) = 1$ whenever $1 \le i < j \le n$.

EXAMPLE 13 Determine whether the integers 10, 17, and 21 are pairwise relatively prime and whether the integers 10, 19, and 24 are pairwise relatively prime.

Solution: Since $\gcd(10, 17) = 1$, $\gcd(10, 21) = 1$, and $\gcd(17, 21) = 1$, we conclude that 10, 17, and 21 are pairwise relatively prime.

Since $\gcd(10, 24) = 2 > 1$, we see that 10, 19, and 24 are not pairwise relatively prime. ■

Another way to find the greatest common divisor of two integers is to use the prime factorizations of these integers. Suppose that the prime factorizations of the integers a and b, neither equal to zero, are

$$a = p_1^{a_1} p_2^{a_2} \cdots p_n^{a_n}, \quad b = p_1^{b_1} p_2^{b_2} \cdots p_n^{b_n},$$

where each exponent is a nonnegative integer, and where all primes occurring in the prime factorization of either a or b are included in both factorizations, with zero exponents if necessary. Then $\gcd(a, b)$ is given by

$$\gcd(a, b) = p_1^{\min(a_1, b_1)} p_2^{\min(a_2, b_2)} \cdots p_n^{\min(a_n, b_n)},$$

where $\min(x, y)$ represents the minimum of the two numbers x and y. To show that this formula for $\gcd(a, b)$ is valid, we must show that the integer on the right-hand side divides both a and b, and that no larger integer also does. This integer does divide both a and b, since the power of each prime in the factorization does not exceed the power of this prime in either the factorization of a or that of b. Further, no larger integer can divide both a and b, because the exponents of the primes in this factorization cannot be increased, and no other primes can be included.

EXAMPLE 14 Since the prime factorizations of 120 and 500 are $120 = 2^3 \cdot 3 \cdot 5$ and $500 = 2^2 \cdot 5^3$, the greatest common divisor is

$$\gcd(120, 500) = 2^{\min(3,2)} 3^{\min(1,0)} 5^{\min(1,3)} = 2^2 3^0 5^1 = 20.$$ ■

Prime factorizations can also be used to find the **least common multiple** of two integers.

DEFINITION 7. The *least common multiple* of the positive integers *a* and *b* is the smallest positive integer that is divisible by both *a* and *b*. The least common multiple of *a* and *b* is denoted by lcm(*a*, *b*).

The least common multiple exists because the set of integers divisible by both a and b is nonempty, and every nonempty set of positive integers has a least element (by the well-ordering property, which will be discussed in Chapter 3). Suppose that the prime factorizations of a and b are as before. Then the least common multiple of a and b is given by

$$\text{lcm}(a, b) = p_1^{\max(a_1, b_1)} p_2^{\max(a_2, b_2)} \cdots p_n^{\max(a_n, b_n)},$$

where $\max(x, y)$ denotes the maximum of the two numbers x and y. This formula is valid since a common multiple of a and b has at least $\max(a_i, b_i)$ factors of p_i in its prime factorization, and the least common multiple has no other prime factors besides those in a and b.

EXAMPLE 15 What is the least common multiple of $2^3 3^5 7^2$ and $2^4 3^3$?

Solution: We have

$$\text{lcm}(2^3 3^5 7^2, 2^4 3^3) = 2^{\max(3,4)} 3^{\max(5,3)} 7^{\max(2,0)} = 2^4 3^5 7^2. \qquad \blacksquare$$

The following theorem gives the relationship between the greatest common divisor and least common multiple of two integers. It can be proved using the formulae we have derived for these quantities. The proof of this theorem is left as an exercise for the reader.

THEOREM 5 **Let *a* and *b* be positive integers. Then**

$$ab = \gcd(a, b) \cdot \text{lcm}(a, b).$$

MODULAR ARITHMETIC

In some situations we care only about the remainder of an integer when it is divided by some specified positive integer. For instance, when we ask what time it will be (on a 24-hour clock) 50 hours from now, we care only about the remainder when 50 plus the current hour is divided by 24. Since we are often interested only in remainders, we have special notations for them.

DEFINITION 8. Let *a* be an integer and *m* be a positive integer. We denote by *a* mod *m* the remainder when *a* is divided by *m*.

It follows from the definition of remainder that a **mod** m is the integer r such that $a = qm + r$ and $0 \leq r < m$.

EXAMPLE 16 We see that 17 **mod** 5 = 2, −133 **mod** 9 = 2, and 2001 **mod** 101 = 82. ■

We also have a notation to indicate that two integers have the same remainder when they are divided by the positive integer m.

DEFINITION 9. If a and b are integers and m is a positive integer, then a is *congruent to b modulo m* if m divides $a - b$. We use the notation $a \equiv b \pmod{m}$ to indicate that a is congruent to b modulo m. If a and b are not congruent modulo m, we write $a \not\equiv b \pmod{m}$.

Note that $a \equiv b \pmod{m}$ if and only if a **mod** m = b **mod** m.

EXAMPLE 17 Determine whether 17 is congruent to 5 modulo 6 and whether 24 and 14 are congruent modulo 6.

Solution: Since 6 divides $17 - 5 = 12$, we see that $17 \equiv 5 \pmod{6}$. However, since $24 - 14 = 10$ is not divisible by 6, we see that $24 \not\equiv 14 \pmod{6}$. ■

The great German mathematician Karl Friedrich Gauss developed the concept of congruences at the end of the eighteenth century.

The notion of congruences has played an important role in the development of number theory. The following theorem provides a useful way to work with congruences.

---

web

Karl Friedrich Gauss (1777–1855). Karl Friedrich Gauss, the son of a bricklayer, was a child prodigy. He demonstrated his potential at the age of 10, when he quickly solved a problem assigned by a teacher to keep the class busy. The teacher asked the students to find the sum of the first 100 positive integers. Gauss realized that this sum could be found by forming 50 pairs, each with the sum 101: 1+100, 2+99, …, 50+51. This brilliance attracted the sponsorship of patrons, including Duke Ferdinand of Brunswick, who made it possible for Gauss to attend Caroline College and the University of Göttingen. While a student, he invented the method of least squares, which is used to estimate the most likely value of a variable from experimental results. In 1796 Gauss made a fundamental discovery in geometry, advancing a subject that had not advanced since ancient times. He showed that a 17-sided regular polygon could be drawn using just a ruler and compass.

In 1799 Gauss presented the first rigorous proof of the Fundamental Theorem of Arithmetic, which states that a polynomial of degree n has exactly n roots (counting multiplicities). Gauss achieved worldwide fame when he successfully calculated the orbit of the first asteroid discovered, Ceres, using scanty data.

Gauss was called the Prince of Mathematics by his contemporary mathematicians. Although Gauss is noted for his many discoveries in geometry, algebra, analysis, astronomy, and physics, he had a special interest in number theory, which can be seen from his statement "Mathematics is the queen of the sciences, and the theory of numbers is the queen of mathematics." Gauss laid the foundations for modern number theory with the publication of his book *Disquisitiones Arithmeticae* in 1801.

THEOREM 6 **Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that $a = b + km$.**

Proof: If $a \equiv b \pmod{m}$, then $m \mid (a - b)$. This means that there is an integer k such that $a - b = km$, so that $a = b + km$. Conversely, if there is an integer k such that $a = b + km$, then $km = a - b$. Hence, m divides $a - b$, so that $a \equiv b \pmod{m}$. □

The following theorem shows how congruences work with respect to addition and multiplication.

THEOREM 7 **Let m be a positive integer. If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then**

$$a + c \equiv b + d \pmod{m} \qquad \text{and} \qquad ac \equiv bd \pmod{m}.$$

Proof: Since $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, there are integers s and t with $b = a + sm$ and $d = c + tm$. Hence,

$$b + d = (a + sm) + (c + tm) = (a + c) + m(s + t)$$

and

$$bd = (a + sm)(c + tm) = ac + m(at + cs + stm).$$

Hence,

$$a + c \equiv b + d \pmod{m} \qquad \text{and} \qquad ac \equiv bd \pmod{m}.$$ □

EXAMPLE 18 Since $7 \equiv 2 \pmod{5}$ and $11 \equiv 1 \pmod{5}$, it follows from Theorem 7 that

$$18 = 7 + 11 \equiv 2 + 1 = 3 \pmod{5}$$

and that

$$77 = 7 \cdot 11 \equiv 2 \cdot 1 = 2 \pmod{5}.$$ ■

APPLICATIONS OF CONGRUENCES

Number theory has applications to a wide range of areas. We will introduce three applications in this section: the use of congruences to assign memory locations to computer files, the generation of pseudorandom numbers, and cryptosystems based on modular arithmetic.

EXAMPLE 19 Hashing Functions The central computer at your school maintains records for each student. How can memory locations be assigned so that student records can be retrieved quickly? The solution to this problem is to use a suitably chosen **hashing function.** Records are identified using a **key,** which uniquely identifies each student's records. For instance, student records are often identified using the Social Security number of the student as the key. A hashing function h assigns memory location $h(k)$ to the record that has k as its key.

In practice, many different hashing functions are used. One of the most common is the function

$$h(k) = k \ \mathbf{mod} \ m$$

where m is the number of available memory locations.

Hashing functions should be easily evaluated so that files can be quickly located. The hashing function $h(k) = k \ \mathbf{mod} \ m$ meets this requirement; to find $h(k)$, we need only compute the remainder when k is divided by m. Furthermore, the hashing function should be onto, so that all memory locations are possible. The function $h(k) = k \ \mathbf{mod} \ m$ also satisfies this property.

For example, when $m = 111$, the record of the student with Social Security number 064212848 is assigned to memory location 14, since

$$h(064212848) = 064212848 \ \mathbf{mod} \ 111 = 14.$$

Similarly, since

$$h(037149212) = 037149212 \ \mathbf{mod} \ 111 = 65,$$

the record of the student with Social Security number 037149212 is assigned to memory location 65.

Since a hashing function is not one-to-one (since there are more possible keys than memory locations), more than one file may be assigned to a memory location. When this happens, we say that a **collision** occurs. One way to resolve a collision is to assign the first free location following the occupied memory location assigned by the hashing function. For example, after making the two earlier assignments, we assign location 15 to the record of the student with the Social Security number 107405723. To see this, first note that $h(k)$ maps this Social Security number to location 14, since

$$h(107405723) = 107405723 \ \mathbf{mod} \ 111 = 14,$$

but this location is already occupied (by the file of the student with Social Security number 064212848). However, memory location 15, the first location following memory location 14, is free.

There are many more sophisticated ways to resolve collisions that are more efficient than the simple method we have described. These are discussed in the references on hashing functions given at the end of the book. ■

EXAMPLE 20 Pseudorandom Numbers Randomly chosen numbers are often needed for computer simulations. Different methods have been devised for generating numbers that have properties of randomly chosen numbers. Because numbers generated by systematic methods are not truly random, they are called **pseudorandom numbers.**

The most commonly used procedure for generating pseudorandom numbers is the **linear congruential method.** We choose four integers: the **modulus** m, **multiplier** a, **increment** c, and **seed** x_0, with $2 \le a < m$, $0 \le c < m$, and $0 \le x_0 < m$. We generate a sequence of pseudorandom numbers $\{x_n\}$, with $0 \le x_n < m$ for all n, by successively using the congruence

$$x_{n+1} = (ax_n + c) \ \mathbf{mod} \ m.$$

(This is an example of a recursive definition, discussed in Section 3.3. In that section we will show that such sequences are well defined.)

Many computer experiments require the generation of pseudorandom numbers between 0 and 1. To generate such numbers, we divide numbers generated with a linear congruential generator by the modulus: that is, we use the numbers x_n/m.

For instance, the sequence of pseudorandom numbers generated by choosing $m = 9$, $a = 7$, $c = 4$, and $x_0 = 3$, can be found as follows:

$$\begin{aligned}
x_1 &= 7x_0 + 4 = 7 \cdot 3 + 4 = 25 \textbf{ mod } 9 = 7,\\
x_2 &= 7x_1 + 4 = 7 \cdot 7 + 4 = 53 \textbf{ mod } 9 = 8,\\
x_3 &= 7x_2 + 4 = 7 \cdot 8 + 4 = 60 \textbf{ mod } 9 = 6,\\
x_4 &= 7x_3 + 4 = 7 \cdot 6 + 4 = 46 \textbf{ mod } 9 = 1,\\
x_5 &= 7x_4 + 4 = 7 \cdot 1 + 4 = 11 \textbf{ mod } 9 = 2,\\
x_6 &= 7x_5 + 4 = 7 \cdot 2 + 4 = 18 \textbf{ mod } 9 = 0,\\
x_7 &= 7x_6 + 4 = 7 \cdot 0 + 4 = 4 \textbf{ mod } 9 = 4,\\
x_8 &= 7x_7 + 4 = 7 \cdot 4 + 4 = 32 \textbf{ mod } 9 = 5,\\
x_9 &= 7x_8 + 4 = 7 \cdot 5 + 4 = 39 \textbf{ mod } 9 = 3.
\end{aligned}$$

Since $x_9 = x_0$ and since each term depends only on the previous term, the following sequence is generated:

$$3, 7, 8, 6, 1, 2, 0, 4, 5, 3, 7, 8, 6, 1, 2, 0, 4, 5, 3, \ldots .$$

This sequence contains nine different numbers before repeating.

Most computers do use linear congruential generators to generate pseudorandom numbers. Often, a linear congruential generator with increment $c = 0$ is used. Such a generator is called a **pure multiplicative generator.** For example, the pure multiplicative generator with modulus $2^{31} - 1$ and multiplier $7^5 = 16{,}807$ is widely used. With these values, it can be shown that $2^{31} - 2$ numbers are generated before repetition begins. ■

CRYPTOLOGY

Congruences have many applications to discrete mathematics and computer science. Discussions of these applications can be found in the suggested readings given at the end of the book. One of the most important applications of congruences involves **cryptology,** which is the study of secret messages. One of the earliest known uses of cryptology was by Julius Caesar. He made messages secret by shifting each letter three letters forward in the alphabet (sending the last three letters of the alphabet to the first three). For instance, using this scheme the letter B is sent to E and the letter X is sent to A. This is an example of **encryption,** that is, the process of making a message secret.

To express Caesar's encryption process mathematically, first replace each letter by an integer from 0 to 25, based on its position in the alphabet. For example, replace A by 0, K by 10, and Z by 25. Caesar's encryption method can be represented by the function f that assigns to the nonnegative integer p, $p \leq 25$, the integer $f(p)$ in the set $\{0, 1, 2, \ldots, 25\}$ with

$$f(p) = (p + 3) \textbf{ mod } 26.$$

In the encrypted version of the message, the letter represented by p is replaced with the letter represented by $(p + 3)$ **mod** 26.

EXAMPLE 21 What is the secret message produced from the message "MEET YOU IN THE PARK" using the Caesar cipher?

Solution: First replace the letters in the message with numbers. This produces

$$12\ 4\ 4\ 19 \qquad 24\ 14\ 20 \qquad 8\ 13 \qquad 19\ 7\ 4 \qquad 15\ 0\ 17\ 10.$$

Now replace each of these numbers p by $f(p) = (p + 3)$ **mod** 26. This gives

15 7 7 22 1 17 23 11 16 22 10 7 18 3 20 13.

Translating this back to letters produces the encrypted message "PHHW BRX LQ WKH SDUN." ■

To recover the original message from a secret message encrypted by the Caesar cipher, the function f^{-1}, the inverse of f, is used. Note that the function f^{-1} sends an integer p from $\{0, 1, 2, \ldots, 25\}$ to $f^{-1}(p) = (p - 3)$ **mod** 26. In other words, to find the original message, each letter is shifted back three letters in the alphabet, with the first three letters sent to the last three letters of the alphabet. The process of determining the original message from the encrypted message is called **decryption.**

There are various ways to generalize the Caesar cipher. For example, instead of shifting each letter by 3, we can shift each letter by k, so that

$$f(p) = (p + k) \textbf{ mod } 26.$$

Such a cipher is called a **shift cipher.** Note that decryption can be carried out using

$$f^{-1}(p) = (p - k) \textbf{ mod } 26.$$

Obviously, Caesar's method and shift ciphers do not provide a high level of security. There are various ways to enhance this method. One approach that slightly enhances the security is to use a function of the form

$$f(p) = (ap + b) \textbf{ mod } 26,$$

where a and b are integers, chosen such that f is a bijection. (Such a mapping is called an *affine transformation.*) This provides a number of possible encryption systems. The use of one of these systems is illustrated in the following example.

EXAMPLE 22 What letter replaces the letter K when the function $f(p) = (7p + 3)$ **mod** 26 is used for encryption?

Solution: First, note that 10 represents K. Then, using the encryption function specified, it follows that $f(10) = (7 \cdot 10 + 3)$ **mod** $26 = 21$. Since 21 represents V, K is replaced by V in the encrypted message. ■

Caesar's encryption method, and the generalization of this method, proceed by replacing each letter of the alphabet by another letter in the alphabet. Encryption methods of this kind are vulnerable to attacks based on the frequency of occurrence of letters in the message. More sophisticated encryption methods are based on replacing blocks of letters with other blocks of letters. There are a number of techniques based on modular arithmetic for encrypting blocks of letters. A discussion of these can be found in the suggested readings listed at the end of the book.

Exercises

1. Does 17 divide each of the following numbers?
   a) 68 b) 84 c) 357 d) 1001

2. Show that if a is an integer other than 0, then
   a) 1 divides a. b) a divides 0.

3. Show that part (2) of Theorem 1 is true.

4. Show that part (3) of Theorem 1 is true.

5. Show that if $a \mid b$ and $b \mid a$, where a and b are integers, then $a = b$ or $a = -b$.

6. Show that if a, b, c, and d are integers such that $a \mid c$ and $b \mid d$, then $ab \mid cd$.

7. Show that if a, b, and c are integers such that $ac \mid bc$, then $a \mid b$.

8. Are the following integers primes?
a) 19 **b)** 27 **c)** 93
d) 101 **e)** 107 **f)** 113

9. In each of the following cases, what are the quotient and remainder?
a) 19 is divided by 7 **b)** -111 is divided by 11
c) 789 is divided by 23 **d)** 1001 is divided by 13
e) 0 is divided by 19 **f)** 3 is divided by 5
g) -1 is divided by 3 **h)** 4 is divided by 1

10. Find the prime factorization of each of the following.
a) 39 **b)** 81 **c)** 101
d) 143 **e)** 289 **f)** 899

11. Find the prime factorization of 10!.

***12.** How many zeros are there at the end of 100!?

***13.** An **irrational number** is a real number x that cannot be written as the ratio of two integers. Show that $\log_2 3$ is an irrational number.

14. Which positive integers less than 12 are relatively prime to 12?

15. Determine whether the following sets of integers are pairwise relatively prime.
a) (11, 15, 19) **b)** (14, 15, 21)
c) (12, 17, 31, 37) **d)** (7, 8, 9, 11)

16. We call a positive integer **perfect** if it equals the sum of its positive divisors other than itself.
a) Show that 6 and 28 are perfect.
b) Show that $2^{p-1}(2^p - 1)$ is a perfect number when $2^p - 1$ is prime.

17. Let m be a positive integer. Show that $a \equiv b \pmod{m}$ if $a \mathbf{\ mod\ } m = b \mathbf{\ mod\ } m$.

18. Let m be a positive integer. Show that $a \mathbf{\ mod\ } m = b \mathbf{\ mod\ } m$ if $a \equiv b \pmod{m}$.

19. Show that if $2^n - 1$ is prime, then n is prime. [*Hint:* Use the identity $2^{ab} - 1 = (2^a - 1)\cdot(2^{a(b-1)} + 2^{a(b-2)} + \cdots + 2^a + 1)$.]

20. Determine whether each of the following integers is prime, verifying some of Mersenne's claims.
a) $2^7 - 1$ **b)** $2^9 - 1$
c) $2^{11} - 1$ **d)** $2^{13} - 1$

21. The value of the **Euler ϕ-function** at the positive integer n is defined to be the number of positive integers less than or equal to n that are relatively prime to n. (*Note:* ϕ is the Greek letter phi.) Find
a) $\phi(4)$. **b)** $\phi(10)$. **c)** $\phi(13)$.

22. Show that n is prime if and only if $\phi(n) = n - 1$.

23. What is the value of $\phi(p^k)$ when p is prime and k is a positive integer?

24. What are the greatest common divisors of the following pairs of integers?
a) $2^2 \cdot 3^3 \cdot 5^5, 2^5 \cdot 3^3 \cdot 5^2$
b) $2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13, 2^{11} \cdot 3^9 \cdot 11 \cdot 17^{14}$
c) $17, 17^{17}$
d) $2^2 \cdot 7, 5^3 \cdot 13$
e) $0, 5$
f) $2 \cdot 3 \cdot 5 \cdot 7, 2 \cdot 3 \cdot 5 \cdot 7$

***25.** Show that if n and k are positive integers, then $\lceil n/k \rceil = \lfloor (n-1)/k \rfloor + 1$.

26. Show that if a is an integer and d is a positive integer greater than 1, then the quotient and remainder obtained when a is divided by d are $\lfloor a/d \rfloor$ and $a - d\lfloor a/d \rfloor$, respectively.

27. Find a formula for the integer with smallest absolute value that is congruent to an integer a modulo m, where m is a positive integer.

28. Evaluate the following quantities.
a) -17 **mod** 2 **b)** 144 **mod** 7
c) -101 **mod** 13 **d)** 199 **mod** 19

29. Evaluate the following quantities.
a) 13 **mod** 3 **b)** -97 **mod** 11
c) 155 **mod** 19 **d)** -221 **mod** 23

30. List five integers that are congruent to 4 modulo 12.

31. Decide whether each of the following integers is congruent to 5 modulo 17.
a) 80 **b)** 103 **c)** -29 **d)** -122

32. If the product of two integers is $2^7 3^8 5^2 7^{11}$ and their greatest common divisor is $2^3 3^4 5$, what is their least common multiple?

33. Show that if a and b are positive integers then $ab = \gcd(a, b) \cdot \operatorname{lcm}(a, b)$. [*Hint:* Use the prime factorizations of a and b and the formulae for $\gcd(a, b)$ and $\operatorname{lcm}(a, b)$ in terms of these factorizations.]

34. Show that if $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, where a, b, c, d, and m are integers with $m \ge 2$, then $a - c \equiv b - d \pmod{m}$.

35. Show that if $n \mid m$, where n and m are positive integers greater than 1, and if $a \equiv b \pmod{m}$, where a and b are integers, then $a \equiv b \pmod{n}$.

36. Show that if a, b, c, and m are integers such that $m \ge 2$, $c > 0$, and $a \equiv b \pmod{m}$, then $ac \equiv bc \pmod{mc}$.

37. Show that $ac \equiv bc \pmod{m}$, where a, b, c, and m are integers with $m \ge 2$, does not necessarily imply that $a \equiv b \pmod{m}$.

38. Show that if a, b, and m are integers such that $m \ge 2$ and $a \equiv b \pmod{m}$, then $\gcd(a, m) = \gcd(b, m)$.

39. Show that if a, b, k, and m are integers such that $k \ge 1$, $m \ge 2$, and $a \equiv b \pmod{m}$, then $a^k \equiv b^k \pmod{m}$ whenever k is a positive integer.

40. Which memory locations are assigned by the hashing function $h(k) = k$ **mod** 101 to the records of students with the following Social Security numbers?
a) 104578690 **b)** 432222187
c) 372201919 **d)** 501338753

41. A parking lot has 31 visitor spaces, numbered from 0 to 30. Visitors are assigned parking spaces using the hashing function $h(k) = k \textbf{ mod } 31$, where k is the number formed from the first three digits on a visitor's license plate.

a) Which spaces are assigned by the hashing function to cars that have the following first three digits on their license plates?

317, 918, 007, 100, 111, 310

b) Describe a procedure visitors should follow to find a free parking space, when the space they are assigned is occupied.

42. What sequence of pseudorandom numbers is generated using the linear congruential generator $x_{n+1} = (4x_n + 1) \textbf{ mod } 7$ with seed $x_0 = 3$?

43. What sequence of pseudorandom numbers is generated using the pure multiplicative generator $x_{n+1} = 3x_n \textbf{ mod } 11$ with seed $x_0 = 2$?

44. Write an algorithm in pseudocode for generating a sequence of pseudorandom numbers using a linear congruential generator.

45. Encrypt the message "DO NOT PASS GO" by translating the letters into numbers, applying the encryption function given, and then translating the numbers back into letters.

a) $f(p) = (p + 3) \textbf{ mod } 26$ (the Caesar cipher)

b) $f(p) = (p + 13) \textbf{ mod } 26$

c) $f(p) = (3p + 7) \textbf{ mod } 26$

46. Decrypt the following messages encrypted using the Caesar cipher.

a) EOXH MHDQV

b) WHVW WRGDB

c) HDW GLP VXP

Books are identified by an **International Standard Book Number (ISBN)**, a 10-digit code $x_1 x_2 \ldots x_{10}$, assigned by the publisher. These 10 digits consist of blocks identifying the language, the publisher, the number assigned to the book by its publishing company, and finally, a 1-digit check digit that is either a digit or the letter X (used to represent 10). This check digit is selected so that $\sum_{i=1}^{10} ix_i \equiv 0 \pmod{11}$ and is used to detect errors in individual digits and transposition of digits.

47. The first nine digits of the ISBN of the third edition of this book are 0-07-053965. What is the check digit for this book?

48. The ISBN of *Elementary Number Theory and Its Applications*, 3d ed., is 0-201-57Q89-1, where Q is a digit. Find the value of Q.

49. Determine whether the check digit of the ISBN for this textbook was computed correctly by the publisher.

50. Find the smallest positive integer with exactly n different factors when n is

a) 3. **b)** 4. **c)** 5. **d)** 6. **e)** 10.

51. Can you find a formula or rule for the nth term of a sequence related to the prime numbers or prime factorizations so that the initial terms of the sequence have the following values?

a) 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, …

b) 1, 2, 3, 2, 5, 2, 7, 2, 3, 2, 11, 2, 13, 2, …

c) 1, 2, 2, 3, 2, 4, 2, 4, 3, 4, 2, 6, 2, 4, …

d) 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, …

e) 1, 2, 3, 3, 5, 5, 7, 7, 7, 7, 11, 11, 13, 13, …

f) 1, 2, 6, 30, 210, 2310, 30030, 510510, 9699690, 223092870, …

52. Can you find a formula or rule for the nth term of a sequence related to the prime numbers or prime factorizations so that the initial terms of the sequence have the following values?

a) 2, 2, 3, 5, 5, 7, 7, 11, 11, 11, 11, 13, 13, …

b) 0, 1, 2, 2, 3, 3, 4, 4, 4, 4, 5, 5, 6, 6, …

c) 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, …

d) 1, −1, −1, 0, −1, 1, −1, 0, 0, 1, −1, 0, −1, 1, 1, …

e) 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, …

f) 4, 9, 25, 49, 121, 169, 289, 361, 529, 841, 961, 1369, …

2.4

Integers and Algorithms

INTRODUCTION

As mentioned in Section 2.1, the term *algorithm* originally referred to procedures for performing arithmetic operations using the decimal representations of integers. These algorithms, adapted for use with binary representations, are the basis for computer arithmetic. They provide good illustrations of the concept of an algorithm and the complexity of algorithms. For these reasons, they will be discussed in this section.

There are many important algorithms involving integers besides those used in arithmetic. We will begin our discussion of integers and algorithms with the Euclidean algorithm. It is one of the most useful algorithms, and perhaps the oldest algorithm in mathematics. We will also describe an algorithm for finding the base b expansion of a positive integer for any base b.

THE EUCLIDEAN ALGORITHM

The method described in Section 2.3 for computing the greatest common divisor of two integers, using the prime factorizations of these integers, is inefficient. The reason is that it is time-consuming to find prime factorizations. We will give a more efficient method of finding the greatest common divisor, called the **Euclidean algorithm.** This algorithm has been known since ancient times. It is named after the ancient Greek mathematician Euclid, who included a description of this algorithm in his *Elements*.

Before describing the Euclidean algorithm, we will show how it is used to find gcd(91, 287). First, divide 287, the larger of the two integers, by 91, the smaller, to obtain

$$287 = 91 \cdot 3 + 14.$$

Any divisor of 91 and 287 must also be a divisor of $287 - 91 \cdot 3 = 14$. Also, any divisor of 91 and 14 must also be a divisor of $287 = 91 \cdot 3 + 14$. Hence, the greatest common divisor of 91 and 287 is the same as the greatest common divisor of 91 and 14. This means that the problem of finding gcd(91, 287) has been reduced to the problem of finding gcd(91, 14).

Next, divide 91 by 14 to obtain

$$91 = 14 \cdot 6 + 7.$$

Since any common divisor of 91 and 14 also divides $91 - 14 \cdot 6 = 7$ and any common divisor of 14 and 7 divides 91, it follows that $\gcd(91, 14) = \gcd(14, 7)$.

Continue by dividing 14 by 7, to obtain

$$14 = 7 \cdot 2.$$

Since 7 divides 14, it follows that $\gcd(14, 7) = 7$, and since $\gcd(287, 91) = \gcd(91, 14) = \gcd(14, 7) = 7$, the original problem has been solved.

We now describe how the Euclidean algorithm works in generality. We will use successive divisions to reduce the problem of finding the greatest common divisor of two positive integers to the same problem with smaller integers, until one of the integers is zero.

The Euclidean algorithm is based on the following result about greatest common divisors and the division algorithm.

Euclid (c. 350 B.C.E.). Euclid was the author of the most successful mathematics book ever written, the *Elements*, which appeared in over 1000 different editions from ancient to modern times. Little is known about Euclid's life, other than that he taught at the famous academy at Alexandria. Apparently, Euclid did not stress applications. When a student asked what he would get by learning geometry, Euclid explained that knowledge was worth acquiring for its own sake and told his servant to give the student a coin "since he must make a profit from what he learns."

LEMMA 1 **Let $a = bq + r$, where a, b, q, and r are integers. Then $\gcd(a, b) = \gcd(b, r)$.**

Proof: If we can show that the common divisors of a and b are the same as the common divisors of b and r, we will have shown that $\gcd(a, b) = \gcd(b, r)$, since both pairs must have the same *greatest* common divisor.

So suppose that d divides both a and b. Then it follows that d also divides $a - bq = r$ (from Theorem 1 of Section 2.3). Hence, any common divisor of a and b is also a common divisor of b and r.

Likewise, suppose that d divides both b and r. Then d also divides $bq + r = a$. Hence, any common divisor of b and r is also a common divisor of a and b.

Consequently, $\gcd(a, b) = \gcd(b, r)$. □

Suppose that a and b are positive integers with $a \geq b$. Let $r_0 = a$ and $r_1 = b$. When we successively apply the division algorithm, we obtain

$$\begin{array}{lll} r_0 &= r_1q_1 + r_2 & 0 \leq r_2 < r_1, \\ r_1 &= r_2q_2 + r_3 & 0 \leq r_3 < r_2, \\ & \cdot & \\ & \cdot & \\ & \cdot & \\ r_{n-2} &= r_{n-1}q_{n-1} + r_n & 0 \leq r_n < r_{n-1}, \\ r_{n-1} &= r_nq_n. & \end{array}$$

Eventually a remainder of zero occurs in this sequence of successive divisions, since the sequence of remainders $a = r_0 > r_1 > r_2 > \cdots \geq 0$ cannot contain more than a terms. Furthermore, it follows from Lemma 1 that

$$\begin{aligned} \gcd(a, b) &= \gcd(r_0, r_1) = \gcd(r_1, r_2) = \cdots = \gcd(r_{n-2}, r_{n-1}) \\ &= \gcd(r_{n-1}, r_n) = \gcd(r_n, 0) = r_n. \end{aligned}$$

Hence, the greatest common divisor is the last nonzero remainder in the sequence of divisions.

EXAMPLE 1 Find the greatest common divisor of 414 and 662 using the Euclidean algorithm.

Solution: Successive uses of the division algorithm give:

$$\begin{aligned} 662 &= 414 \cdot 1 + 248 \\ 414 &= 248 \cdot 1 + 166 \\ 248 &= 166 \cdot 1 + 82 \\ 166 &= 82 \cdot 2 + 2 \\ 82 &= 2 \cdot 41. \end{aligned}$$

Hence, $\gcd(414, 662) = 2$, since 2 is the last nonzero remainder. ■

The Euclidean algorithm is expressed in pseudocode in Algorithm 1.

ALGORITHM 1 **The Euclidean Algorithm.**

```
procedure gcd($a$, $b$: positive integers)
$x := a$
$y := b$
while $y \neq 0$
begin
    $r := x$ mod $y$
    $x := y$
    $y := r$
end {gcd($a$, $b$) is $x$}
```

In Algorithm 1, the initial values of x and y are a and b, respectively. At each stage of the procedure, x is replaced by y, and y is replaced by x **mod** y, which is the remainder when x is divided by y. This process is repeated as long as $y \neq 0$. The algorithm terminates when $y = 0$, and the value of x at that point, the last nonzero remainder in the procedure, is the greatest common divisor of a and b.

We will study the time complexity of the Euclidean algorithm in Section 3.3, where we will show that the number of divisions required to find the greatest common divisor of a and b, where $a \geq b$, is $O(\log b)$.

REPRESENTATIONS OF INTEGERS

In everyday life we use decimal notation to express integers. For example, 965 is used to denote $9 \cdot 10^2 + 6 \cdot 10 + 5$. However, it is often convenient to use bases other than 10. In particular, computers usually use binary notation (with 2 as the base) when carrying out arithmetic, and octal (base 8) or hexadecimal (base 16) notation when expressing characters, such as letters or digits. In fact, we can use any positive integer greater than 1 as the base when expressing integers. This is stated in the following theorem.

THEOREM 1 **Let b be a positive integer greater than 1. Then if n is a positive integer, it can be expressed uniquely in the form**

$$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b + a_0,$$

where k is a nonnegative integer, $a_0, a_1, \ldots, a_k$ are nonnegative integers less than b, and $a_k \neq 0$.

The proof of this theorem can be found in the suggested readings referred to at the end of the book. The representation of n given in Theorem 1 is called the **base b expansion of n.** The base b expansion of n is denoted by $(a_k a_{k-1} \cdots a_1 a_0)_b$. For instance, $(245)_8$ represents $2 \cdot 8^2 + 4 \cdot 8 + 5 = 165$.

Choosing 2 as the base gives **binary expansions** of integers. In binary notation each digit is either a 0 or a 1. In other words, the binary expansion of an integer is just a bit string. Binary expansions (and related expansions that are variants of binary expansions) are used by computers to represent and do arithmetic with integers.

EXAMPLE 2 What is the decimal expansion of the integer that has $(101011111)_2$ as its binary expansion?

Solution: We have

$$(101011111)_2 = 2^8 + 2^6 + 2^4 + 2^3 + 2^2 + 2 + 1 = 351. \qquad \blacksquare$$

Sixteen is another base used in computer science. The base 16 expansion of an integer is called its **hexadecimal** expansion. Sixteen different digits are required for such expansions. Usually, the hexadecimal digits used are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F, where the letters A through F represent the digits corresponding to the numbers 10 through 15 (in decimal notation).

EXAMPLE 3 What is the decimal expansion of the hexadecimal expansion of $(2AE0B)_{16}$?

Solution: We have

$$(2AE0B)_{16} = 2 \cdot 16^4 + 10 \cdot 16^3 + 14 \cdot 16^2 + 0 \cdot 16 + 11 = (175627)_{10}. \qquad \blacksquare$$

Since a hexadecimal digit is represented using four bits, **bytes,** which are bit strings of length eight, can be represented by two hexadecimal digits. For instance, $(11100101)_2 = (E5)_{16}$ since $(1110)_2 = (E)_{16}$ and $(0101)_2 = (5)_{16}$.

We will now describe an algorithm for constructing the base b expansion of an integer n. First, divide n by b to obtain a quotient and remainder, that is,

$$n = bq_0 + a_0, \qquad 0 \le a_0 < b.$$

The remainder, a_0, is the rightmost digit in the base b expansion of n. Next, divide q_0 by b to obtain

$$q_0 = bq_1 + a_1, \qquad 0 \le a_1 < b.$$

We see that a_1 is the second digit from the right in the base b expansion of n. Continue this process, successively dividing the quotients by b, obtaining additional base b digits as the remainders. This process terminates when we obtain a quotient equal to zero.

EXAMPLE 4 Find the base 8 expansion of $(12345)_{10}$.

Solution: First, divide 12345 by 8 to obtain

$$12345 = 8 \cdot 1543 + 1.$$

Successively dividing quotients by 8 gives

$$\begin{aligned} 1543 &= 8 \cdot 192 + 7 \\ 192 &= 8 \cdot 24 + 0 \\ 24 &= 8 \cdot 3 + 0 \\ 3 &= 8 \cdot 0 + 3. \end{aligned}$$

Since the remainders are the digits of the base 8 expansion of 12345, it follows that

$$(12345)_{10} = (30071)_8. \quad \blacksquare$$

The pseudocode given in Algorithm 2 finds the base b expansion $(a_{k-1} \cdots a_1 a_0)_b$ of the integer n.

ALGORITHM 2 **Constructing Base b Expansions.**

```
procedure base b expansion(n: positive integer)
q := n
k := 0
while q ≠ 0
begin
    a_k := q mod b
    q := ⌊q/b⌋
    k := k + 1
end {the base b expansion of n is (a_{k-1} ··· a_1 a_0)_b}
```

In Algorithm 2, q represents the quotient obtained by successive divisions by b, starting with $q = n$. The digits in the base b expansion are the remainders of these divisions and are given by q **mod** b. The algorithm terminates when a quotient $q = 0$ is reached.

ALGORITHMS FOR INTEGER OPERATIONS

The algorithms for performing operations with integers using their binary expansions are extremely important in computer arithmetic. We will describe algorithms for the addition and the multiplication of two integers expressed in binary notation. We will also analyze the computational complexity of these algorithms, in terms of the actual number of bit operations used. Throughout this discussion, suppose that the binary expansions of a and b are

$$a = (a_{n-1}a_{n-2} \cdots a_1 a_0)_2, \quad b = (b_{n-1}b_{n-2} \cdots b_1 b_0)_2,$$

so that a and b each have n bits (putting bits equal to 0 at the beginning of one of these expansions if necessary).

Consider the problem of adding two integers in binary notation. A procedure to perform addition can be based on the usual method for adding numbers with pencil and paper. This method proceeds by adding pairs of binary digits together with carries, when they occur, to compute the sum of two integers. This procedure will now be specified in detail.

To add a and b, first add their rightmost bits. This gives

$$a_0 + b_0 = c_0 \cdot 2 + s_0,$$

where s_0 is the rightmost bit in the binary expansion of $a + b$ and c_0 is the **carry**, which is either 0 or 1. Then add the next pair of bits and the carry,

$$a_1 + b_1 + c_0 = c_1 \cdot 2 + s_1,$$

where s_1 is the next bit (from the right) in the binary expansion of $a + b$, and c_1 is the carry. Continue this process, adding the corresponding bits in the two binary expansions and the carry, to determine the next bit from the right in the binary expansion of $a + b$.

At the last stage, add a_{n-1}, b_{n-1}, and c_{n-2} to obtain $c_{n-1} \cdot 2 + s_{n-1}$. The leading bit of the sum is $s_n = c_{n-1}$. This procedure produces the binary expansion of the sum, namely, $a + b = (s_n s_{n-1} s_{n-2} \cdots s_1 s_0)_2$.

EXAMPLE 5 Add $a = (1110)_2$ and $b = (1011)_2$.

Solution: Following the procedure specified in the algorithm, first note that

$$a_0 + b_0 = 0 + 1 = 0 \cdot 2 + 1,$$

so that $c_0 = 0$ and $s_0 = 1$. Then, since

$$a_1 + b_1 + c_0 = 1 + 1 + 0 = 1 \cdot 2 + 0,$$

it follows that $c_1 = 1$ and $s_1 = 0$. Continuing,

$$a_2 + b_2 + c_1 = 1 + 0 + 1 = 1 \cdot 2 + 0,$$

so that $c_2 = 1$ and $s_2 = 0$. Finally, since

$$a_3 + b_3 + c_2 = 1 + 1 + 1 = 1 \cdot 2 + 1,$$

it follows that $c_3 = 1$ and $s_3 = 1$. This means that $s_4 = c_3 = 1$. Therefore, $s = a + b = (11001)_2$. This addition is displayed in Figure 1. ■

$$\begin{array}{r} \scriptstyle 1\ 1\ \ \ \ \ \\ 1\ 1\ 1\ 0 \\ 1\ 0\ 1\ 1 \\ \hline 1\ 1\ 0\ 0\ 1 \end{array}$$

FIGURE 1 Adding $(1110)_2$ and $(1011)_2$.

The algorithm for addition can be described using pseudocode as follows.

ALGORITHM 3 Addition of Integers.

```
procedure add(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}a_{n-2} \cdots a_1a_0)_2$
    and $(b_{n-1}b_{n-2} \cdots b_1b_0)_2$, respectively}
$c := 0$
for $j := 0$ to $n - 1$
begin
    $d := \lfloor (a_j + b_j + c)/2 \rfloor$
    $s_j := a_j + b_j + c - 2d$
    $c := d$
end
$s_n := c$
{the binary expansion of the sum is $(s_n s_{n-1} \cdots s_0)_2$}
```

Next, the number of additions of bits used by Algorithm 3 will be analyzed.

EXAMPLE 6 How many additions of bits are required to use Algorithm 3 to add two integers with n bits (or less) in their binary representations?

Solution: Two integers are added by successively adding pairs of bits and, when it occurs, a carry. Adding each pair of bits and the carry requires three or fewer additions of bits. Thus, the total number of additions of bits used is less than three times the number of bits in the expansion. Hence, the number of additions of bits used by Algorithm 3 to add two n-bit integers is $O(n)$. ■

Next, consider the multiplication of two n-bit integers a and b. The conventional algorithm (used when multiplying with pencil and paper) works as follows. Using the distributive law, we see that

$$ab = a\sum_{j=0}^{n-1} b_j 2^j = \sum_{j=0}^{n-1} a(b_j 2^j).$$

We can compute ab using this equation. We first note that $ab_j = a$ if $b_j = 1$ and $ab_j = 0$ if $b_j = 0$. Each time we multiply a term by 2, we shift its binary expansion one place to the left and add a zero at the tail end of the expansion. Consequently, we can obtain $(ab_j)2^j$ by **shifting** the binary expansion of ab_j j places to the left, adding j zero bits at the tail end of this binary expansion. Finally, we obtain ab by adding the n integers $ab_j 2^j$, $j = 0, 1, 2, \ldots, n-1$.

The following example illustrates the use of this algorithm.

EXAMPLE 7 Find the product of $a = (110)_2$ and $b = (101)_2$.

Solution: First note that

$$ab_0 \cdot 2^0 = (110)_2 \cdot 1 \cdot 2^0 = (110)_2,$$

$$ab_1 \cdot 2^1 = (110)_2 \cdot 0 \cdot 2^1 = (0000)_2,$$

and

$$ab_2 \cdot 2^2 = (110)_2 \cdot 1 \cdot 2^2 = (11000)_2.$$

To find the product, add $(110)_2$, $(0000)_2$, and $(11000)_2$. Carrying out these additions (using Algorithm 3, including initial zero bits when necessary) shows that $ab = (11110)_2$. This multiplication is displayed in Figure 2. ■

$$\begin{array}{r} 1\,1\,0 \\ 1\,0\,1 \\ \hline 1\,1\,0 \\ 0\,0\,0\;\; \\ 1\,1\,0\;\;\;\; \\ \hline 1\,1\,1\,1\,0 \end{array}$$

FIGURE 2 Multiplying $(110)_2$ and $(101)_2$.

This procedure for multiplication can be described using pseudocode as follows.

ALGORITHM 4 **Multiplying Integers.**

```
procedure multiply(a, b: positive integers)
{the binary expansions of $a$ and $b$ are $(a_{n-1}a_{n-2}\cdots a_1a_0)_2$
    and $(b_{n-1}b_{n-2}\cdots b_1b_0)_2$, respectively}
for $j := 0$ to $n - 1$
begin
    if $b_j = 1$ then $c_j :=$ $a$ shifted $j$ places
    else $c_j := 0$
end
{$c_0, c_1, \ldots, c_{n-1}$ are the partial products}
$p := 0$
for $j := 0$ to $n - 1$
    $p := p + c_j$
{$p$ is the value of $ab$}
```

Next, we determine the number of additions of bits and shifts of bits used by Algorithm 4 to multiply two integers.

EXAMPLE 8 How many additions of bits and shifts of bits are used to multiply a and b using Algorithm 4?

Solution: Algorithm 4 computes the products of a and b by adding the partial products $c_0, c_1, c_2, \ldots,$ and c_{n-1}. When $b_j = 1$, we compute the partial product c_j by shifting the binary expansion of a j bits. When $b_j = 0$, no shifts are required since $c_j = 0$. Hence, to find all n of the integers $ab_j2^j, j = 0, 1, \ldots, n-1$, requires at most

$$0 + 1 + 2 + \cdots + n - 1$$

shifts. Hence, by Example 4 in Section 1.8 the number of shifts required is $O(n^2)$.

To add the integers ab_j from $j = 0$ to $j = n - 1$ requires the addition of an n-bit integer, an $(n + 1)$-bit integer, ..., and a $(2n)$-bit integer. We know from Example 8 that each of these additions requires $O(n)$ additions of bits. Consequently, a total of $O(n^2)$ additions of bits are required for all n additions. ■

Surprisingly, there are more efficient algorithms than the conventional algorithm for multiplying integers. One such algorithm, which uses $O(n^{1.585})$ bit operations to multiply n-bit numbers, will be described in Chapter 5.

Exercises

1. Use the Euclidean algorithm to find
 a) gcd(12, 18). **b)** gcd(111, 201).
 c) gcd(1001, 1331). **d)** gcd(12345, 54321).

2. Use the Euclidean algorithm to find
 a) gcd(1, 5) **b)** gcd(100, 101)
 c) gcd(123, 277) **d)** gcd(1529, 14039)
 e) gcd(1529, 14038) **f)** gcd(11111, 111111)

3. How many divisions are required to find gcd(21, 34) using the Euclidean algorithm?

4. How many divisions are required to find gcd(34, 55) using the Euclidean algorithm?

5. Convert the following integers from decimal notation to binary notation.
 a) 231 **b)** 4532 **c)** 97644

6. Convert the following integers from decimal notation to binary notation.
 a) 321 **b)** 1023 **c)** 100632

7. Convert the following integers from binary notation to decimal notation.
 a) 1 1111
 b) 10 0000 0001
 c) 1 0101 0101
 d) 110 1001 0001 0000

8. Convert the following integers from binary notation to decimal notation.
 a) 1 1011 **b)** 10 1011 0101
 c) 11 1011 1110 **d)** 111 1100 0001 1111

9. Devise a simple method for converting from hexadecimal notation to binary notation.

10. Devise a simple method for converting from binary notation to hexadecimal notation.

11. Convert each of the following integers from hexadecimal notation to binary notation.
 a) 80E **b)** 135AB
 c) ABBA **d)** DEFACED

12. Convert each of the following integers from binary notation to hexadecimal notation.
 a) 1111 0111 **b)** 1010 1010 1010
 c) 111 0111 0111 0111

13. Show that every positive integer can be represented uniquely as the sum of distinct powers of 2. (*Hint:* Consider binary expansions of integers.)

14. It can be shown that every integer can be uniquely represented in the form

$$e_k3^k + e_{k-1}3^{k-1} + \cdots + e_13 + e_0,$$

where $e_j = -1, 0,$ or 1 for $j = 0, 1, 2, \ldots, k$. Expansions of this type are called **balanced ternary expansions.** Find the balanced ternary expansions of

a) 5 **b)** 13 **c)** 37 **d)** 79

15. Show that a positive integer is divisible by 3 if and only if the sum of its decimal digits is divisible by 3.

16. Show that a positive integer is divisible by 11 if and only if the difference of the sum of its decimal digits in even-numbered positions and the sum of its decimal digits in odd-numbered positions is divisible by 11.

17. Show that a positive integer is divisible by 3 if and only if the difference of the sum of its binary digits in even-numbered positions and the sum of its binary digits in odd-numbered positions is divisible by 3.

One's complement representations of integers are used to simplify computer arithmetic. To represent positive and negative integers with absolute value less than 2^{n-1}, a total of n bits is used. The leftmost bit is used to represent the sign. A 0 bit in this position is used for positive integers, and a 1 bit in this position is used for negative integers. For positive integers the remaining bits are identical to the binary expansion of the integer. For negative integers, the remaining bits are obtained by first finding the binary expansion of the absolute value of the integer, and then taking the complement of each of these bits, where the complement of a 1 is a 0 and the complement of a 0 is a 1.

18. Find the one's complement representations, using bit strings of length six, of the following integers.

a) 22 **b)** 31 **c)** -7 **d)** -19

19. What integer does each of the following one's complement representations of length five represent?

a) 11001 **b)** 01101 **c)** 10001 **d)** 11111

20. If m is a positive integer less than 2^{n-1}, how is the one's complement representation of $-m$ obtained from the one's complement of m, when bit strings of length n are used?

21. How is the one's complement representation of the sum of two integers obtained from the one's complement representations of these integers?

22. How is the one's complement representation of the difference of two integers obtained from the one's complement representations of these integers?

23. Show that the integer m with one's complement representation $(a_{n-1}a_{n-2}\cdots a_1a_0)$ can be found using the equation $m = -a_{n-1}(2^{n-1} - 1) + \sum_{i=0}^{n-2} a_i2^i$.

Two's complement representations of integers are also used to simplify computer arithmetic and are used more commonly than one's complement representations. To represent an integer x with $-2^{n-1} \le x \le 2^{n-1} - 1$ for a specified positive integer n, a total of n bits is used. The leftmost bit is used to represent the sign. A 0 bit in this position is used for positive integers, and a 1 bit in this position is used for negative integers, just as in one's complement expansions. For a positive integer, the remaining bits are identical to the binary expansion of the integer. For a negative integer, the remaining bits are the bits of the binary expansion of $2^{n-1} - |x|$. Two's complement expansions of integers are often used by computers because addition and subtraction of integers can be performed easily using these expansions, where these integers can be either positive or negative.

24. Answer Exercise 18, but this time find the two's complement expansion using bit strings of length six.

25. Answer Exercise 19 if each expansion is a two's complement expansion of length five.

26. Answer Exercise 20 for two's complement expansions.

27. Answer Exercise 21 for two's complement expansions.

28. Answer Exercise 22 for two's complement expansions.

29. Show that the integer m with two's complement representation $(a_{n-1}a_{n-2}\cdots a_1a_0)$ can be found using the equation $m = -a_{n-1} \cdot 2^{n-1} + \sum_{i=0}^{n-2} a_i2^i$.

30. Give a simple algorithm for forming the two's complement representation of an integer from its one's complement representation.

31. Sometimes integers are encoded by using four-digit binary expansions to represent each decimal digit. This produces the **binary coded decimal** form of the integer. For instance, 791 is encoded in this way by 011110010001. How many bits are required to represent a number with n decimal digits using this type of encoding?

A **Cantor expansion** is a sum of the form

$$a_nn! + a_{n-1}(n-1)! + \cdots + a_22! + a_11!,$$

where a_i is an integer with $0 \le a_i \le i$ for $i = 1, 2, \ldots, n$.

32. Find the Cantor expansions of

a) 2 **b)** 7 **c)** 19

d) 87 **e)** 1000 **f)** 1,000,000

***33.** Describe an algorithm that finds the Cantor expansion of an integer.

***34.** Describe an algorithm to add two integers from their Cantor expansions.

35. Add $(10111)_2$ and $(11010)_2$ by working through each step of the algorithm for addition given in the text.

36. Multiply $(1110)_2$ and $(1010)_2$ by working through each step of the algorithm for multiplication given in the text.

37. Describe an algorithm for finding the difference of two binary expansions.

38. Estimate the number of bit operations used to subtract two binary expansions.

39. Devise an algorithm that, given the binary expansions of the integers a and b, determines whether $a > b$, $a = b$, or $a < b$.

40. How many bit operations does the comparison algorithm from Exercise 39 use when the larger of a and b has n bits in its binary expansion?

41. Estimate the complexity of Algorithm 2 for finding the base b expansion of an integer n in terms of the number of divisions used.

2.5

Applications of Number Theory

INTRODUCTION

Number theory has many applications, especially to computer science. In Section 2.3 we described several of these applications, including hashing functions, the generation of pseudorandom numbers, and shift ciphers. This section continues our introduction to number theory, developing some key results and presenting two important applications: a method for performing arithmetic with large integers and a recently invented type of cryptosystem, called a *public key system*. In such a cryptosystem, we do not have to keep encryption keys secret, since knowledge of an encryption key does not help someone decrypt messages in a realistic amount of time. Privately held decryption keys are used to decrypt messages.

Before developing these applications, we will introduce some key results that play a central role in number theory and its applications. For example, we will show how to solve systems of linear congruences modulo pairwise relatively prime integers using the Chinese Remainder Theorem, and then show how to use this result as a basis for performing arithmetic with large integers. We will introduce Fermat's Little Theorem and the concept of a pseudoprime and will show how to use these concepts to develop a public key cryptosystem.

SOME USEFUL RESULTS

An important result we will use throughout this section is that the greatest common divisor of two integers a and b can be expressed in the form

$$sa + tb,$$

where s and t are integers. In other words, $\gcd(a, b)$ can be expressed as a **linear combination** with integer coefficients of a and b. For example, $\gcd(6, 14) = 2$, and $2 = (-2) \cdot 6 + 1 \cdot 14$. We state this fact as Theorem 1.

THEOREM 1 **If a and b are positive integers, then there exist integers s and t such that $\gcd(a, b) = sa + tb$.**

We will not give a formal proof of Theorem 1, but we will provide an example of a method for finding a linear combination of two integers equal to their greatest common divisor. (In this section, we will assume that a linear combination has integer coefficients.) The method proceeds by working backward through the divisions of the Euclidean algorithm.

EXAMPLE 1 Express $\gcd(252, 198) = 18$ as a linear combination of 252 and 198.

Solution: To show that $\gcd(252, 198) = 18$, the Euclidean algorithm uses the following divisions:

$$\begin{aligned} 252 &= 1 \cdot 198 + 54 \\ 198 &= 3 \cdot 54 + 36 \\ 54 &= 1 \cdot 36 + 18 \\ 36 &= 2 \cdot 18. \end{aligned}$$

Using the next-to-last division (the third division), we can express $\gcd(252, 198) = 18$ as a linear combination of 54 and 36. We find that

$$18 = 54 - 1 \cdot 36.$$

The second division tells us that

$$36 = 198 - 3 \cdot 54.$$

Substituting this expression for 36 into the previous equation, we can express 18 as a linear combination of 54 and 198. We have

$$18 = 54 - 1 \cdot 36 = 54 - 1 \cdot (198 - 3 \cdot 54) = 4 \cdot 54 - 1 \cdot 198.$$

The first division tells us that

$$54 = 252 - 1 \cdot 198.$$

Substituting this expression for 54 into the previous equation, we can express 18 as a linear combination of 252 and 198. We conclude that

$$18 = 4 \cdot (252 - 1 \cdot 198) - 1 \cdot 198 = 4 \cdot 252 - 5 \cdot 198,$$

completing the solution. ■

The method used in Example 1 works for any pair of positive integers. [There are more efficient methods for expressing $\gcd(a, b)$ as a linear combination of a and b; consult the references at the end of the book to learn more about these methods.]

We will use Theorem 1 to develop several useful results. One of our goals will be to prove the part of the Fundamental Theorem of Arithmetic asserting that a positive integer has at most one prime factorization. We will show that if a positive integer has a factorization into primes, where the primes are written in nondecreasing order, then this factorization is unique.

First, we need to develop some results about divisibility.

LEMMA 1 **If a, b, and c are positive integers such that $\gcd(a, b) = 1$ and $a \mid bc$, then $a \mid c$.**

Proof: Since $\gcd(a, b) = 1$, by Theorem 1 there are integers s and t such that

$$sa + tb = 1.$$

Multiplying both sides of this equation by c, we obtain

$$sac + tbc = c.$$

Using Theorem 1 of Section 2.3, we can use this last equation to show that $a \mid c$. By part 2 of that theorem, $a \mid tbc$. Since $a \mid sac$ and $a \mid tbc$, by part 1 of that theorem, we conclude that a divides $sac + tbc$, and hence $a \mid c$. This finishes the proof. ☐

We will use the following generalization of Lemma 1 in the proof of uniqueness of prime factorizations. (The proof of Lemma 2 is left as an exercise in Section 3.2, since it can be most easily carried out using the method of mathematical induction, which will be covered in that section.)

LEMMA 2 **If p is a prime and $p \mid a_1a_2\cdots a_n$ where each a_i is an integer, then $p \mid a_i$ for some i.**

We can now show that a factorization of an integer into primes is unique. That is, we will show that every integer can be written as the product of primes in nondecreasing order in at most one way. This is part of the Fundamental Theorem of Arithmetic. We will prove the other part, that every integer has a factorization into primes, in Section 3.2.

Proof (of the uniqueness of the prime factorization of a positive integer): Suppose that the positive integer n can be written as the product of primes in two different ways, say, $n = p_1p_2\cdots p_s$ and $n = q_1q_2\cdots q_t$, each p_i and q_j are primes such that $p_1 \le p_2 \le \cdots \le p_s$ and $q_1 \le q_2 \le \cdots \le q_t$.

When we remove all common primes from the two factorizations, we have

$$p_{i_1}p_{i_2}\cdots p_{i_u} = q_{j_1}q_{j_2}\cdots q_{j_v},$$

where no prime occurs on both sides of this equation and u and v are positive integers. By Lemma 2 it follows that p_{i_1} divides q_{j_k} for some k. Since no prime divides another prime, this is impossible. Consequently, there can be at most one factorization of n into primes in nondecreasing order. ☐

Lemma 1 can also be used to prove a result about dividing both sides of a congruence by the same integer. We have shown (Theorem 7 in Section 2.3) that we can multiply both sides of a congruence by the same integer. However, dividing both sides of a congruence by an integer does not always produce a valid congruence, as the following example shows.

EXAMPLE 2 The congruence $14 \equiv 8 \pmod{6}$ holds, but both sides of this congruence cannot be divided by 2 since $14/2 = 7$ and $8/2 = 4$, but $7 \not\equiv 4 \pmod{6}$. ■

However, using Lemma 1, we can show that we can divide both sides of a congruence by an integer relatively prime to the modulus. This is stated as Theorem 2.

THEOREM 2 **Let m be a positive integer and let a, b, and c be integers. If $ac \equiv bc \pmod{m}$ and $\gcd(c, m) = 1$, then $a \equiv b \pmod{m}$.**

Proof: Since $ac \equiv bc \pmod{m}$, $m \mid ac - bc = c(a-b)$. By Lemma 1, since $\gcd(c, m) = 1$, it follows that $m \mid a - b$. We conclude that $a \equiv b \pmod{m}$. ☐

LINEAR CONGRUENCES

A congruence of the form

$$ax \equiv b \pmod{m}$$

where m is a positive integer, a and b are integers, and x is a variable, is called a **linear congruence.** Such congruences arise throughout number theory and its applications.

How can we solve the linear congruence $ax \equiv b \pmod{m}$, that is, find all integers x that satisfy this congruence? One method that we will describe uses an integer $\bar{a}$ such that $\bar{a}a \equiv 1 \pmod{m}$, if such an integer exists. Such an integer $\bar{a}$ is said to be an **inverse** of a modulo m. Theorem 3 guarantees that an inverse of a modulo m exists whenever a and m are relatively prime.

THEOREM 3 **If a and m are relatively prime integers and $m > 1$, then an inverse of a modulo m exists. Furthermore, this inverse is unique modulo m. (That is, there is a unique positive integer $\bar{a}$ less than m that is an inverse of a modulo m and every other inverse of a modulo m is congruent to $\bar{a}$ modulo m.)**

Proof: By Theorem 1, since $\gcd(a, m) = 1$, there are integers s and t such that

$$sa + tm = 1.$$

This implies that

$$sa + tm \equiv 1 \pmod{m}.$$

Since $tm \equiv 0 \pmod{m}$, it follows that

$$sa \equiv 1 \pmod{m}.$$

Consequently, s is an inverse of a modulo m. That this inverse is unique modulo m is left as Exercise 9 at the end of this section. ☐

The proof of Theorem 3 describes a method for finding the inverse of a modulo m when a and m are relatively prime: find a linear combination of a and m that equals 1 (which can be done by working backward through the steps of the Euclidean algorithm); the coefficient of a in this linear combination is an inverse of a modulo m. We illustrate this procedure in Example 3.

EXAMPLE 3 Find an inverse of 3 modulo 7.

Solution: Since $\gcd(3, 7) = 1$, Theorem 2 tells us that an inverse of 3 modulo 7 exists. The Euclidean algorithm ends quickly when used to find the greatest common divisor of 3 and 7:

$$7 = 2 \cdot 3 + 1.$$

From this equation we see that

$$-2 \cdot 3 + 1 \cdot 7 = 1.$$

This shows that -2 is an inverse of 3 modulo 7. (Note that every integer congruent to -2 modulo 7 is also an inverse of 3, such as 5, -9, 12, and so on.) ■

When we have an inverse $\overline{a}$ of a modulo m, we can easily solve the congruence $ax \equiv b \pmod{m}$ by multiplying both sides of the linear congruence by $\overline{a}$, as Example 4 illustrates.

EXAMPLE 4 What are the solutions of the linear congruence $3x \equiv 4 \pmod{7}$?

Solution: By Example 3 we know that -2 is an inverse of 3 modulo 7. Multiplying both sides of the congruence by -2 shows that

$$-2 \cdot 3x \equiv -2 \cdot 4 \pmod{7}.$$

Since $-6 \equiv 1 \pmod{7}$ and $-8 \equiv 6 \pmod{7}$, it follows that if x is a solution, then $x \equiv -8 \equiv 6 \pmod{7}$.

We need to determine whether every x with $x \equiv 6 \pmod{7}$ is a solution. Assume that $x \equiv 6 \pmod{7}$. Then, by Theorem 7 of Section 2.3, it follows that

$$3x \equiv 3 \cdot 6 = 18 \equiv 4 \pmod{7},$$

which shows that all such x satisfy the congruence. We conclude that the solutions to the congruence are the integers x such that $x \equiv 6 \pmod{7}$, namely, $6, 13, 20, \ldots$ and $-1, -8, -15, \ldots$. ■

THE CHINESE REMAINDER THEOREM

Systems of linear congruences arise in many contexts. For example, as we will see later, they are the basis for a method that can be used to perform arithmetic with large integers. Such systems can even be found as word puzzles in the writings of ancient Chinese and Hindu mathematicians, such as that given in Example 5.

EXAMPLE 5 In the first century, the Chinese mathematician Sun-Tsu asked:

> There are certain things whose number is unknown. When divided by 3, the remainder is 2; when divided by 5, the remainder is 3; and when divided by 7, the remainder is 2. What will be the number of things?

This puzzle can be translated into the following question: What are the solutions of the systems of congruences

$$x \equiv 2 \pmod{3},$$

$$x \equiv 3 \pmod{5},$$

$$x \equiv 2 \pmod{7}?$$

We will solve this system, and with it Sun-Tsu's puzzle, later in this section. ■

The *Chinese Remainder Theorem*, named after the Chinese heritage of problems involving systems of linear congruences, states that when the moduli of a system of linear congruences are pairwise relatively prime, there is a unique solution of the system modulo the product of the moduli.

THEOREM 4 **THE CHINESE REMAINDER THEOREM** **Let $m_1, m_2, \ldots, m_n$ be pairwise relatively prime positive integers. The system**

$$\begin{aligned} x &\equiv a_1 \pmod{m_1},\\ x &\equiv a_2 \pmod{m_2},\\ &\;\;\vdots\\ x &\equiv a_n \pmod{m_n} \end{aligned}$$

has a unique solution modulo $m = m_1 m_2 \cdots m_n$. (That is, there is a solution x with $0 \le x < m$, and all other solutions are congruent modulo m to this solution.)

Proof: To establish this theorem, we need to show that a solution exists and that it is unique modulo m. We will show that a solution exists by describing a way to construct this solution; showing that the solution is unique modulo m is Exercise 20 at the end of this section.

To construct a simultaneous solution, first let

$$M_k = m/m_k$$

for $k = 1, 2, \ldots, n$. That is, M_k is the product of the moduli except for m_k. Since m_i and m_k have no common factors greater than 1 when $i \neq k$, it follows that $\gcd(m_k, M_k) = 1$. Consequently, by Theorem 3, we know that there is an integer y_k, an inverse of M_k modulo m_k, such that

$$M_k y_k \equiv 1 \pmod{m_k}.$$

To construct a simultaneous solution, form the sum

$$x = a_1 M_1 y_1 + a_2 M_2 y_2 + \cdots + a_n M_n y_n.$$

We will now show that x is a simultaneous solution. First, note that since $M_j \equiv 0 \pmod{m_k}$ whenever $j \neq k$, all terms except the kth term in this sum are congruent to 0 modulo m_k. Since $M_k y_k \equiv 1 \pmod{m_k}$ we see that

$$x \equiv a_k M_k y_k \equiv a_k \pmod{m_k},$$

for $k = 1, 2, \ldots, n$. We have shown that x is a simultaneous solution to the n congruences. □

The following example illustrates how to use the construction given in the proof of Theorem 4 to solve a system of congruences. We will solve the system given in Example 5, arising in Sun-Tsu's puzzle.

EXAMPLE 6 To solve the system of congruences in Example 5, first let $m = 3 \cdot 5 \cdot 7 = 105$, $M_1 = m/3 = 35$, $M_2 = m/5 = 21$, and $M_3 = m/7 = 15$. We see that 2 is an inverse of $M_1 = 35$ modulo 3, since $35 \equiv 2 \pmod 3$; 1 is an inverse of $M_2 = 21$ modulo 5, since $21 \equiv 1 \pmod 5$; and 1 is an inverse of $M_3 = 15 \pmod 7$, since $15 \equiv 1 \pmod 7$. The solutions to this system are those x such that

$$\begin{aligned} x \equiv a_1 M_1 y_1 + a_2 M_2 y_2 + a_3 M_3 y_3 &= 2 \cdot 35 \cdot 2 + 3 \cdot 21 \cdot 1 + 2 \cdot 15 \cdot 1 \pmod{105}\\ &= 233 \equiv 23 \pmod{105}. \end{aligned}$$

It follows that 23 is the smallest positive integer that is a simultaneous solution. We conclude that 23 is the smallest positive integer that leaves a remainder of 2 when divided by 3, a remainder of 3 when divided by 5, and a remainder of 2 when divided by 7. ■

COMPUTER ARITHMETIC WITH LARGE INTEGERS

Suppose that $m_1, m_2, \ldots, m_n$ are pairwise relatively prime integers greater than or equal to 2 and let m be their product. By the Chinese Remainder Theorem, we can show (see Exercise 18) that an integer a with $0 \leq a < m$ can be uniquely represented by the n-tuple consisting of its remainders upon division by m_i, $i = 1, 2, \ldots, n$. That is, we can uniquely represent a by

$$(a \mathbf{\ mod\ } m_1, a \mathbf{\ mod\ } m_2, \ldots, a \mathbf{\ mod\ } m_n).$$

EXAMPLE 7 What are the pairs used to represent the nonnegative integers less than 12 when they are represented by the ordered pair where the first component is the remainder of the integer upon division by 3 and the second component is the remainder of the integer upon division by 4?

Solution: We have the following representations, obtained by finding the remainder of each integer when it is divided by 3 and by 4:

$$\begin{array}{lll} 0 = (0,0) & 4 = (1,0) & 8 = (2,0) \\ 1 = (1,1) & 5 = (2,1) & 9 = (0,1) \\ 2 = (2,2) & 6 = (0,2) & 10 = (1,2) \\ 3 = (0,3) & 7 = (1,3) & 11 = (2,3). \end{array}$$

■

To perform arithmetic with large integers, we select moduli $m_1, m_2, \ldots, m_n$, where each m_i is an integer greater than 2, $\gcd(m_i, m_j) = 1$ whenever $i \neq j$, and $m = m_1 m_2 \cdots m_n$ is greater than the result of the arithmetic operations we want to carry out.

Once we have selected our moduli, we carry out arithmetic operations with large integers by performing componentwise operations on the n-tuples representing these integers using their remainders upon division by m_i, $i = 1, 2, \ldots, n$. Once we have computed the value of each component in the result, we recover its value by solving a system of n congruences modulo m_i, $i = 1, 2, \ldots, n$. This method of performing arithmetic with large integers has several valuable features. First, it can be used to perform arithmetic with integers larger than can ordinarily be carried out on a computer. Second, computations with respect to the different moduli can be done in parallel, speeding up the arithmetic.

EXAMPLE 8 Suppose that performing arithmetic with integers less than 100 on a certain processor is much quicker than doing arithmetic with larger integers. We can restrict almost all our computations to integers less than 100 if we represent integers using their remainders modulo pairwise relatively prime integers less than 100. For example, we can use the moduli of 99, 98, 97, and 95. (These integers are relatively prime pairwise, since no two have a common factor greater than 1.)

By the Chinese Remainder Theorem, every nonnegative integer less than $99 \cdot 98 \cdot 97 \cdot 95 = 89{,}403{,}930$ can be represented uniquely by its remainders when divided by these four moduli. For example, we represent 123,684 as (33, 8, 9, 89), since $123{,}684 \ \mathbf{mod}\ 99 = 33$, $123{,}684 \ \mathbf{mod}\ 98 = 8$, $123{,}684 \ \mathbf{mod}\ 97 = 9$, and $123{,}684 \ \mathbf{mod}\ 95 = 89$. Similarly, we represent 413,456 as (32, 92, 42, 16).

To find the sum of 123,684 and 413,456, we work with these 4-tuples instead of these two integers directly. We add the 4-tuples componentwise and reduce each component with respect to the appropriate modulus. This yields

$$\begin{aligned}(33, 8, 9, 89) + (32, 92, 42, 16)\\ &= (65 \ \mathbf{mod}\ 99, 100 \ \mathbf{mod}\ 98, 51 \ \mathbf{mod}\ 97, 105 \ \mathbf{mod}\ 95)\\ &= (65, 2, 51, 10).\end{aligned}$$

To find the sum, that is, the integer represented by (65, 2, 51, 10), we need to solve the system of congruences

$$x \equiv 65 \pmod{99}$$

$$x \equiv 2 \pmod{98}$$

$$x \equiv 51 \pmod{97}$$

$$x \equiv 10 \pmod{95}$$

It can be shown (see Exercise 29) that 537,140 is the unique nonnegative solution of this system less than 89,403,930. Consequently, 537,140 is the sum. Note that it is only when we have to recover the integer represented by (65, 2, 51, 10) that we have to do arithmetic with integers larger than 100. ■

Particularly good choices for moduli for arithmetic with large integers are sets of integers of the form $2^k - 1$, where k is a positive integer, since it is easy to do binary arithmetic modulo such integers, and since it is easy to find sets of such integers that are pairwise relatively prime. [The second reason is a consequence of the fact that $\gcd(2^a - 1, 2^b - 1) = 2^{\gcd(a,b)} - 1$, as Exercise 31 shows.] Suppose, for instance, that we can do arithmetic with integers less than 2^{35} easily on our computer, but that working with larger integers requires special procedures. We can use pairwise relatively prime moduli less than 2^{35} to perform arithmetic with integers as large as their product. For example, as Exercise 32 shows, the integers $2^{35} - 1$, $2^{34} - 1$, $2^{33} - 1$, $2^{31} - 1$, $2^{29} - 1$, and $2^{23} - 1$ are pairwise relatively prime. Since the product of these six moduli exceeds 2^{184}, we can perform arithmetic with integers as large as 2^{184} (as long as the results do not exceed this number) by doing arithmetic modulo for each of these six moduli, none of which exceeds 2^{35}.

PSEUDOPRIMES

In Section 2.3 we showed that an integer n is prime when it is not divisible by any prime p with $p \leq \sqrt{n}$. Unfortunately, using this criterion to show that a given integer is prime is inefficient. It requires that we find all primes not exceeding $\sqrt{n}$ and that we carry out trial division by each such prime to see whether it divides n.

Are there more efficient ways to determine whether an integer is prime? Ancient Chinese mathematicians believed that n was prime if and only if

$$2^{n-1} \equiv 1 \pmod{n}.$$

If this were true, it would provide an efficient primality test. Why did they believe this congruence could be used to determine whether an integer is prime? First, they observed that the congruence holds whenever n is prime. For example, 5 is prime and

$$2^{5-1} = 2^4 = 16 \equiv 1 \pmod{5}.$$

Second, they never found a composite integer n for which the congruence holds. The ancient Chinese were only partially correct. They were correct in thinking that the congruence holds whenever n is prime, but they were incorrect in concluding that n is necessarily prime if the congruence holds.

The great French mathematician Fermat showed that the congruence holds when n is prime. He proved the following, more general result.

THEOREM 5 **FERMAT'S LITTLE THEOREM If p is prime and a is an integer not divisible by p, then**

$$a^{p-1} \equiv 1 \pmod{p}.$$

Furthermore, for every integer a we have

$$a^p \equiv a \pmod{p}.$$

The proof of Theorem 5 is outlined in Exercise 17 at the end of this section.

Unfortunately, there are composite integers n such that $2^{n-1} \equiv 1 \pmod{n}$. Such integers are called **pseudoprimes.**

EXAMPLE 9 The integer 341 is a pseudoprime since it is composite ($341 = 11 \cdot 31$) and as Exercise 23 shows

$$2^{340} \equiv 1 \pmod{341}. \qquad \blacksquare$$

Although the ancient Chinese were wrong, pseudoprimes are relatively rare. Their scarcity—and the even greater scarcity of integers that pass more delicate tests that begin by determining whether an integer is a pseudoprime—can be used as the basis for efficient **probabilistic primality tests.** Such tests can be used to quickly show that it almost certainly is the case that a given integer is prime. (More precisely, these tests

---

web **Pierre de Fermat (1601–1665).** Pierre de Fermat, one of the most important mathematicians of the seventeenth century, was a lawyer by profession. He is the most famous amateur mathematician in history. Fermat published little of his mathematical discoveries. It is through his correspondence with other mathematicians that we know of his work. Fermat was one of the inventors of analytic geometry and developed some of the fundamental ideas of calculus. Fermat, along with Pascal, gave probability theory a mathematical basis. Fermat formulated what is now the most famous unsolved problem in mathematics. He asserted that the equation $x^n + y^n = z^n$ has no nontrivial positive integer solutions when n is an integer greater than 2. For more than 300 years, no proof (or counterexample) was found. In his copy of the works of the ancient Greek mathematician Diophantus, Fermat wrote that he had a proof but that it would not fit in the margin. Because the first proof, found by Andrew Wiles in 1994, relies on sophisticated, modern mathematics, most people think that Fermat thought he had a proof, but it was incorrect. However, he may have been tempting others to look for a proof, not being able to find one himself.

show that the probability that an integer that passes a series of tests is prime is close to 1; see Chapter 4 for a discussion of probability.) These probabilistic primality tests can be used, and are used, to find large primes extremely rapidly on computers.

PUBLIC KEY CRYPTOGRAPHY

web

In Section 2.3 we introduced methods for encrypting messages based on congruences. When these encryption methods are used, messages, which are strings of characters, are translated into numbers. Then the number for each character is transformed into another number, either using a shift or an affine transformation modulo 26. These methods are examples of **private key cryptosystems.** Knowing the encryption key lets you quickly find the decryption key. For example, when a shift cipher is used with encryption key k, a number p representing a letter is sent to

$$c = (p + k) \textbf{ mod } 26.$$

Decryption is carried out by shifting by $-k$; that is,

$$p = (c - k) \textbf{ mod } 26.$$

When a private key cryptosystem is used, a pair of people who wish to communicate in secret must have a separate key. Since anyone knowing this key can both encrypt and decrypt messages easily, these two people need to securely exchange the key.

In the mid-1970s, cryptologists introduced the concept of **public key cryptosystems.** When such cryptosystems are used, knowing how to send someone a message does not help you decrypt messages sent to this person. In such a system, every person can have a publicly known encryption key. Only the decryption keys are kept secret, and only the intended recipient of a message can decrypt it, since the encryption key does not let someone find the decryption key without an extraordinary amount of work (such as 2 billion years of computer time).

In 1976, three researchers at M.I.T.—Ron Rivest, Adi Shamir, and Len Adleman—introduced a public key cryptosystem, known as the **RSA system,** from the initials of its inventors. The RSA cryptosystem is based on modular exponentiation modulo of the product of two large primes. Each individual has an encryption key consisting of a modulus $n = pq$, where p and q are large primes, say, with 200 digits each, and an exponent e that is relatively prime to $(p - 1)(q - 1)$. To produce a usable key, two large primes must be found. This can be done quickly on a computer using probabilistic primality tests, referred to earlier in this section. However, the product of these primes $n = pq$, with approximately 400 digits, cannot be factored in a reasonable length of time. As we will see, this is an important reason why decryption cannot be done quickly without a separate decryption key.

RSA ENCRYPTION

In the RSA encryption method, messages are translated into sequences of integers. This can be done by translating each letter into an integer, as is done with the Caesar cipher. These integers are grouped together to form larger integers, each representing a block of letters. The encryption proceeds by transforming the integer M, representing the plaintext (the original message), to an integer C, representing the ciphertext (the encrypted message), using the function

$$C = M^e \textbf{ mod } n.$$

(To perform the encryption, we use an algorithm for fast modular exponentiation, such as that described in Exercise 14 in the Supplementary Exercises at the end of this chapter.) We leave the encrypted message as blocks of numbers and send these to the intended recipient.

Example 10 illustrates how RSA encryption is performed. For practical reasons we use small primes p and q in this example, rather than primes with 100 or more digits. Although the cipher described in this example is not secure, it does illustrate the techniques used in the RSA cipher.

EXAMPLE 10 Encrypt the message STOP using the RSA cryptosystem with $p = 43$ and $q = 59$, so that $n = 43 \cdot 59 = 2537$, and with $e = 13$. Note that

$$\gcd(e, (p-1)(q-1)) = \gcd(13, 42 \cdot 58) = 1.$$

Solution: We translate the letters in STOP into their numerical equivalents and then group the numbers into blocks of four. We obtain

1819 1415.

We encrypt each block using the mapping

$$C = M^{13} \textbf{ mod } 2537.$$

Computations using fast modular multiplication show that $1819^{13} \textbf{ mod } 2537 = 2081$ and $1415^{13} \textbf{ mod } 2537 = 2182$. The encrypted message is 2081 2182. ∎

RSA DECRYPTION

The plaintext message can be quickly recovered when the decryption key d, an inverse of e modulo $(p-1)(q-1)$, is known. [Such an inverse exists since $\gcd(e, (p-1)(q-1)) = 1$.] To see this, note that if $de \equiv 1 \pmod{(p-1)(q-1)}$, there is an integer k such that $de = 1 + k(p-1)(q-1)$. It follows that

$$C^d = (M^e)^d = M^{de} = M^{1+k(p-1)(q-1)}.$$

By Fermat's Little Theorem [assuming that $\gcd(M, p) = \gcd(M, q) = 1$, which holds except in rare cases], it follows that $M^{p-1} \equiv 1 \pmod{p}$ and $M^{q-1} \equiv 1 \pmod{q}$. Consequently,

$$C^d \equiv M \cdot (M^{p-1})^{k(q-1)} \equiv M \cdot 1 \equiv M \pmod{p}$$

and

$$C^d \equiv M \cdot (M^{q-1})^{k(p-1)} \equiv M \cdot 1 \equiv M \pmod{q}.$$

Since $\gcd(p, q) = 1$, it follows by the Chinese Remainder Theorem that

$$C^d \equiv M \pmod{pq}.$$

Example 11 illustrates how to decrypt messages sent using the RSA cryptosystem.

EXAMPLE 11 We receive the encrypted message 0981 0461. What is the decrypted message if it was encrypted using the RSA cipher from Example 10?

Solution: The message was encrypted using the RSA cryptosystem with $n = 43 \cdot 59$ and exponent 13. As Exercise 4 shows, $d = 937$ is an inverse of 13 modulo $42 \cdot 58 = 2436$. We use 937 as our decryption exponent. Consequently, to decrypt a block C, we compute

$$P = C^{937} \mathbf{mod}\ 2537.$$

To decrypt the message, we use the fast modular exponentiation algorithm to compute 0981^{937} **mod** $2537 = 0704$ and 0461^{937} **mod** $2537 = 1115$. Consequently, the numerical version of the original message is 0704 1115. Translating this back to English letters, we see that the message is HELP. ∎

RSA AS A PUBLIC KEY SYSTEM

web Why is the RSA cryptosystem suitable for public key cryptography? When we know the factorization of the modulus n, that is, when we know p and q, we can use the Euclidean algorithm to quickly find an exponent d inverse to e modulo $(p - 1)(q - 1)$. This lets us decrypt messages sent using our key. However, no method is known to decrypt messages that is not based on finding a factorization of n, or that does not also lead to the factorization of n. The most efficient factorization methods known (as of 1999) require billions of years to factor 400-digit integers. Consequently, when p and q are 200-digit primes, messages encrypted using $n = pq$ as the modulus cannot be found in a reasonable time unless the primes p and q are known.

Active research is under way to find new ways to efficiently factor integers. Integers that were thought, as recently as several years ago, to be far too large to be factored in a reasonable amount of time can now be factored routinely. Integers with more than 100 digits, as well as some with more than 150 digits, have been factored using team efforts. When new factorization techniques are found, it will be necessary to use larger primes to ensure secrecy of messages. Unfortunately, messages that were considered secure earlier can be saved and subsequently decrypted by unintended recipients when it becomes feasible to factor the $n = pq$ in the key used for RSA encryption.

The RSA method has been implemented and is used for some particularly sensitive applications. However, the most commonly used cryptosystem is a private key system known as DES (an acronym for the Data Encryption Standard). When DES is used, encryption and decryption can be performed extremely rapidly on a computer. Although some people believe that messages encrypted using DES can be broken by experts, DES is considered sufficiently secure in most situations. The use of public key cryptography, via the RSA system, is growing, but when the RSA system is used, encryption and decryption are too slow (using the current generation of computers) for many applications. However, there are applications that use both private key and public key systems. For example, a public key cryptosystem, such as RSA, can be used to distribute private keys to pairs of individuals when they wish to communicate. These people then use a private key system, such as DES, for encryption and decryption of messages.

Exercises

1. Express the greatest common divisor of each of the following pairs of integers as a linear combination of these integers.
   - **a)** 10, 11
   - **b)** 21, 44
   - **c)** 36, 48
   - **d)** 34, 55
   - **e)** 117, 213
   - **f)** 0, 223
   - **g)** 123, 2347
   - **h)** 3454, 4666
   - **i)** 9999, 11111

2. Express the greatest common divisor of each of the following pairs of integers as a linear combination of these integers.

a) 9, 11 **b)** 33, 44
c) 35, 78 **d)** 21, 55
e) 101, 203 **f)** 124, 323
g) 2002, 2339 **h)** 3457, 4669
i) 10001, 13422

3. Show that 15 is an inverse of 7 modulo 26.

4. Show that 937 is an inverse of 13 modulo 2436.

5. Find an inverse of 4 modulo 9.

6. Find an inverse of 2 modulo 17.

7. Find an inverse of 19 modulo 141.

8. Find an inverse of 144 modulo 233.

***9.** Show that if a and m are relatively prime positive integers, then the inverse of a modulo m is unique modulo m. [*Hint:* Assume that there are two solutions b and c of the congruence $ax \equiv 1 \pmod{m}$. Use Theorem 2 to show that $b \equiv c \pmod{m}$.]

10. Show that an inverse of a modulo m does not exist if $\gcd(a, m) > 1$.

11. Solve the congruence $4x \equiv 5 \pmod{9}$.

12. Solve the congruence $2x \equiv 7 \pmod{17}$.

***13.** Show that if m is a positive integer greater than 1 and $ac \equiv bc \pmod{m}$, then $a \equiv b \bmod m/\gcd(c, m)$.

14. **a)** Show that the positive integers less than 11, except 1 and 10, can be split into pairs of integers such that each pair consists of integers that are inverses of each other modulo 11.

b) Use part (a) to show that $10! \equiv -1 \pmod{11}$.

15. Show that if p is prime, the only solutions of $x^2 \equiv 1 \pmod{p}$ are integers x such that $x \equiv 1 \pmod{p}$ and $x \equiv -1 \pmod{p}$.

***16.** **a)** Generalize the result in part (a) of Exercise 14; that is, show that if p is a prime, the positive integers less than p, except 1 and $p - 1$, can be split into $(p - 3)/2$ pairs of integers such that each pair consists of integers that are inverses of each other. (*Hint:* Use the result of Exercise 15.)

b) From part (a) conclude that $(p - 1)! \equiv -1 \pmod{p}$ whenever p is prime. This result is known as **Wilson's theorem.**

c) What can we conclude if n is a positive integer such that $(n - 1)! \not\equiv -1 \pmod{n}$?

***17.** This exercise outlines a proof of Fermat's Little Theorem.

a) Suppose that a is not divisible by the prime p. Show that no two of the integers $1 \cdot a, 2 \cdot a, \ldots, (p - 1)a$ are congruent modulo p.

b) Conclude from part (a) that the product of $1, 2, \ldots, p - 1$ is congruent modulo p to the product of $a, 2a, \ldots, (p - 1)a$. Use this to show that

$$(p - 1)! \equiv a^{p-1}(p - 1)! \pmod{p}.$$

c) Use Wilson's theorem (proved in Exercise 16) to show that $a^{p-1} \equiv 1 \pmod{p}$ if $p \nmid a$.

d) Use part (c) to show that $a^p \equiv a \pmod{p}$ for all integers a.

18. Use the Chinese Remainder Theorem to show that an integer a, with $0 \le a < m = m_1 m_2 \cdots m_n$, where the integers $m_1, m_2, \ldots, m_n$ are pairwise relatively prime, can be represented uniquely by the n-tuple (a **mod** m_1, a **mod** $m_2, \ldots, a$ **mod** m_n).

***19.** Let $m_1, m_2, \ldots, m_n$ be pairwise relatively prime integers greater than or equal to 2. Show that if $a \equiv b \pmod{m_i}$ for $i = 1, 2, \ldots, n$, then $a \equiv b \pmod{m}$, where $m = m_1 m_2 \cdots m_n$.

***20.** Complete the proof of the Chinese Remainder Theorem by showing that the simultaneous solution of a system of linear congruences modulo pairwise relatively prime integers is unique modulo the product of these moduli. (*Hint:* Assume that x and y are two simultaneous solutions. Show that $m_i \mid x - y$ for all i. Using Exercise 19, conclude that $m = m_1 m_2 \cdots m_n \mid x - y$.)

21. Which integers leave a remainder of 1 when divided by 2 and also leave a remainder of 1 when divided by 3?

22. Which integers are divisible by 5 but leave a remainder of 1 when divided by 3?

23. **a)** Show that $2^{340} \equiv 1 \pmod{11}$ by Fermat's Little Theorem and noting that $2^{340} = (2^{10})^{34}$.

b) Show that $2^{340} \equiv 1 \pmod{31}$ using the fact that $2^{340} = (2^5)^{68} = 32^{68}$.

c) Conclude from parts (a) and (b) that $2^{340} \equiv 1 \pmod{341}$.

24. **a)** Use Fermat's Little Theorem to compute 3^{302} **mod** 5, 3^{302} **mod** 7, and 3^{302} **mod** 11.

b) Use your results from part (a) and the Chinese Remainder Theorem to find 3^{302} **mod** 385. (Note that $385 = 5 \cdot 7 \cdot 11$.)

25. **a)** Use Fermat's Little Theorem to compute 5^{2003} **mod** 7, 5^{2003} **mod** 11, and 5^{2003} **mod** 13.

b) Use your results from part (a) and the Chinese Remainder Theorem to find 5^{2003} **mod** 1001. (Note that $1001 = 7 \cdot 11 \cdot 13$.)

26. Find the nonnegative integer a less than 28 represented by each of the following pairs, where each pair represents (a **mod** 4, a **mod** 7).

a) (0, 0) **b)** (1, 0) **c)** (1, 1)
d) (2, 1) **e)** (2, 2) **f)** (0, 3)
g) (2, 0) **h)** (3, 5) **i)** (3, 6)

27. Express each nonnegative integer a less than 15 using the pair (a **mod** 3, a **mod** 5).

28. Explain how to use the pairs found in Exercise 27 to add 4 and 7.

29. Solve the system of congruences that arises in Example 8.

***30.** Show that if a and b are positive integers, then $(2^a - 1) \bmod (2^b - 1) = 2^{a \bmod b} - 1$.

****31.** Use Exercise 30 to show that if a and b are positive integers, then $\gcd(2^a - 1, 2^b - 1) = 2^{\gcd(a,b)} - 1$. [*Hint:* Show that the remainders obtained when the Euclidean algorithm is used to compute $\gcd(2^a - 1, 2^b - 1)$ are of the form $2^r - 1$, where r is a remainder arising when the Euclidean algorithm is used to find $\gcd(a, b)$.]

32. Use Exercise 31 to show that the integers $2^{35} - 1$, $2^{34} - 1$, $2^{33} - 1$, $2^{31} - 1$, $2^{29} - 1$, and $2^{23} - 1$ are pairwise relatively prime.

33. Show that if p is an odd prime, then every divisor of the Mersenne number $2^p - 1$ is of the form $2kp + 1$, where k is a nonnegative integer. (*Hint:* Use Fermat's Little Theorem and Exercise 31.)

34. Use Exercise 33 to determine whether $M_{13} = 2^{13} - 1 = 8191$ and $M_{23} = 2^{23} - 1 = 8{,}388{,}607$ are prime.

***35.** Show that we can easily factor n when we know that n is the product of two primes, p and q, and we know the value of $(p - 1)(q - 1)$.

36. Encrypt the message ATTACK using the RSA system with $n = 43 \cdot 59$ and $e = 13$, translating each letter into integers and grouping together pairs of integers, as done in Example 10.

37. What is the original message encrypted using the RSA system with $n = 43 \cdot 59$ and $e = 13$ if the encrypted message is 0667 1947 0671? (*Note:* Some computational aid is needed to do this in a realistic amount of time.)

If m is a positive integer, the integer a is a **quadratic residue** of m if $\gcd(a, m) = 1$ and the congruence $x^2 \equiv a \pmod{m}$ has a solution. In other words, a quadratic residue of m is an integer relatively prime to m which is a perfect square modulo m. For example, 2 is a quadratic residue of 7 since $\gcd(2, 7) = 1$ and $3^2 \equiv 2 \pmod{7}$ and 3 is a quadratic nonresidue of 7 since $\gcd(3, 7) = 1$ and $x^2 \equiv 3 \pmod{7}$ has no solution.

38. Which integers are quadratic residues of 11?

39. Show that if p is an odd prime and a is an integer not divisible by p, then the congruence $x^2 \equiv a \pmod{p}$ has either no solutions or exactly two incongruent solutions modulo p.

40. Show that if p is an odd prime, then there are exactly $(p - 1)/2$ quadratic residues of p among the integers $1, 2, \ldots, p - 1$.

If p is an odd prime and a is an integer not divisible by p, the Legendre symbol $\left(\dfrac{a}{p}\right)$ is defined to be 1 if a is a quadratic residue of p and -1 otherwise.

41. Show that if p is an odd prime and a and b are integers with $a \equiv b \pmod{p}$, then

$$\left(\frac{a}{p}\right) = \left(\frac{b}{p}\right).$$

42. Prove that if p is an odd prime and a is a positive integer not divisible by p, then

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod{p}.$$

43. Use Exercise 42 to show that if p is an odd prime and a and b are integers not divisible by p, then

$$\left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right).$$

44. Show that if p is an odd prime, then -1 is a quadratic residue of p if $p \equiv 1 \pmod{4}$ and 1 is not a quadratic residue of p if $p \equiv 3 \pmod{4}$. (*Hint:* Use Exercise 42.)

45. Find all solutions of the congruence $x^2 \equiv 29 \pmod{35}$. (*Hint:* Find the solutions of this congruence modulo 5 and modulo 7, and then use the Chinese Remainder Theorem.)

46. Find all solutions of the congruence $x^2 \equiv 16 \pmod{105}$. (*Hint:* Find the solutions of this congruence modulo 3, modulo 5, and modulo 7, and then use the Chinese Remainder Theorem.)

2.6
Matrices

INTRODUCTION

Matrices are used throughout discrete mathematics to express relationships between elements in sets. In subsequent chapters we will use matrices in a wide variety of models. For instance, matrices will be used in models of communications networks and transportation systems. Many algorithms will be developed that use these matrix models. This section reviews matrix arithmetic that will be used in these algorithms.

DEFINITION 1. A *matrix* is a rectangular array of numbers. A matrix with m rows and n columns is called an $m \times n$ matrix. The plural of matrix is *matrices*. A matrix with the same number of rows as columns is called *square*. Two matrices are *equal* if they have the same number of rows and the same number of columns and the corresponding entries in every position are equal.

EXAMPLE 1 The matrix $\begin{bmatrix} 1 & 1 \\ 0 & 2 \\ 1 & 3 \end{bmatrix}$ is a 3×2 matrix. ■

We now introduce some terminology about matrices. Boldface uppercase letters will be used to represent matrices.

DEFINITION 2. Let

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}.$$

The ith *row* of $\mathbf{A}$ is the $1 \times n$ matrix $[a_{i1}, a_{i2}, \ldots, a_{in}]$. The jth *column* of $\mathbf{A}$ is the $n \times 1$ matrix

$$\begin{bmatrix} a_{1j} \\ a_{2j} \\ \cdot \\ \cdot \\ \cdot \\ a_{nj} \end{bmatrix}.$$

The (i, j)th *element* or *entry* of $\mathbf{A}$ is the element a_{ij}, that is, the number in the ith row and jth column of $\mathbf{A}$. A convenient shorthand notation for expressing the matrix $\mathbf{A}$ is to write $\mathbf{A} = [a_{ij}]$, which indicates that $\mathbf{A}$ is the matrix with its (i, j)th element equal to a_{ij}.

MATRIX ARITHMETIC

The basic operations of matrix arithmetic will now be discussed, beginning with a definition of matrix addition.

DEFINITION 3. Let $\mathbf{A} = [a_{ij}]$ and $\mathbf{B} = [b_{ij}]$ be $m \times n$ matrices. The *sum* of $\mathbf{A}$ and $\mathbf{B}$, denoted by $\mathbf{A} + \mathbf{B}$, is the $m \times n$ matrix that has $a_{ij} + b_{ij}$ as its (i, j)th element. In other words, $\mathbf{A} + \mathbf{B} = [a_{ij} + b_{ij}]$.

The sum of two matrices of the same size is obtained by adding elements in the corresponding positions. Matrices of different sizes cannot be added, since the sum of two matrices is defined only when both matrices have the same number of rows and the same number of columns.

EXAMPLE 2 We have

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 2 & -3 \\ 3 & 4 & 0 \end{bmatrix} + \begin{bmatrix} 3 & 4 & -1 \\ 1 & -3 & 0 \\ -1 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 4 & 4 & -2 \\ 3 & -1 & -3 \\ 2 & 5 & 2 \end{bmatrix}.$$

■

We now discuss matrix products. A product of two matrices is defined only when the number of columns in the first matrix equals the number of rows of the second matrix.

DEFINITION 4. Let $\mathbf{A}$ be an $m \times k$ matrix and $\mathbf{B}$ be a $k \times n$ matrix. The *product* of $\mathbf{A}$ and $\mathbf{B}$, denoted by $\mathbf{AB}$, is the $m \times n$ matrix with (i, j)th entry equal to the sum of the products of the corresponding elements from the ith row of $\mathbf{A}$ and the jth column of $\mathbf{B}$. In other words, if $\mathbf{AB} = [c_{ij}]$, then

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{ik}b_{kj} = \sum_{t=1}^{k} a_{it}b_{tj}.$$

In Figure 1 the colored row of $\mathbf{A}$ and the colored column of $\mathbf{B}$ are used to compute the element c_{ij} of $\mathbf{AB}$. The product of two matrices is not defined when the number of columns in the first matrix and the number of rows in the second matrix are not the same.

We now give some examples of matrix products.

EXAMPLE 3 Let

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 4 \\ 2 & 1 & 1 \\ 3 & 1 & 0 \\ 0 & 2 & 2 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 2 & 4 \\ 1 & 1 \\ 3 & 0 \end{bmatrix}.$$

Find $\mathbf{AB}$ if it is defined.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ik} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mk} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1j} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2j} & \cdots & b_{2n} \\ \vdots & & & \vdots & & \vdots \\ b_{k1} & b_{k2} & \cdots & b_{kj} & \cdots & b_{kn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & & c_{ij} & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{bmatrix}$$

FIGURE 1 The Product of $\mathbf{A} = [a_{ij}]$ and $\mathbf{B} = [b_{ij}]$.

Solution: Since $\mathbf{A}$ is a 4×3 matrix and $\mathbf{B}$ is a 3×2 matrix, the product $\mathbf{AB}$ is defined and is a 4×2 matrix. To find the elements of $\mathbf{AB}$, the corresponding elements of the rows of $\mathbf{A}$ and the columns of $\mathbf{B}$ are first multiplied and then these products are added. For instance, the element in the $(3, 1)$th position of $\mathbf{AB}$ is the sum of the products of the corresponding elements of the third row of $\mathbf{A}$ and the first column of $\mathbf{B}$; namely, $3 \cdot 2 + 1 \cdot 1 + 0 \cdot 3 = 7$. When all the elements of $\mathbf{AB}$ are computed, we see that

$$\mathbf{AB} = \begin{bmatrix} 14 & 4 \\ 8 & 9 \\ 7 & 13 \\ 8 & 2 \end{bmatrix}.$$

■

Matrix multiplication is *not* commutative. That is, if $\mathbf{A}$ and $\mathbf{B}$ are two matrices, it is not necessarily true that $\mathbf{AB}$ and $\mathbf{BA}$ are the same. In fact, it may be that only one of these two products is defined. For instance, if $\mathbf{A}$ is 2×3 and $\mathbf{B}$ is 3×4, then $\mathbf{AB}$ is defined and is 2×4; however, $\mathbf{BA}$ is not defined, since it is impossible to multiply a 3×4 matrix and a 2×3 matrix.

In general, suppose that $\mathbf{A}$ is an $m \times n$ matrix and $\mathbf{B}$ is an $r \times s$ matrix. Then $\mathbf{AB}$ is defined only when $n = r$ and $\mathbf{BA}$ is defined only when $s = m$. Moreover, even when $\mathbf{AB}$ and $\mathbf{BA}$ are both defined, they will not be the same size unless $m = n = r = s$. Hence, if both $\mathbf{AB}$ and $\mathbf{BA}$ are defined and are the same size, then both $\mathbf{A}$ and $\mathbf{B}$ must be square and of the same size. Furthermore, even with $\mathbf{A}$ and $\mathbf{B}$ both $n \times n$ matrices, $\mathbf{AB}$ and $\mathbf{BA}$ are not necessarily equal, as the following example demonstrates.

EXAMPLE 4 Let

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 2 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}.$$

Does $\mathbf{AB} = \mathbf{BA}$?

Solution: We find that

$$\mathbf{AB} = \begin{bmatrix} 3 & 2 \\ 5 & 3 \end{bmatrix} \quad \text{and} \quad \mathbf{BA} = \begin{bmatrix} 4 & 3 \\ 3 & 2 \end{bmatrix}.$$

Hence, $\mathbf{AB} \neq \mathbf{BA}$. ■

ALGORITHMS FOR MATRIX MULTIPLICATION

The definition of the product of two matrices leads to an algorithm that computes the product of two matrices. Suppose that $\mathbf{C} = [c_{ij}]$ is the $m \times n$ matrix that is the product of the $m \times k$ matrix $\mathbf{A} = [a_{ij}]$ and the $k \times n$ matrix $\mathbf{B} = [b_{ij}]$. The algorithm based on the definition of the matrix product is expressed in pseudocode in Algorithm 1.

ALGORITHM 1 Matrix Multiplication.

```
procedure matrix multiplication(A, B: matrices)
for $i$ := 1 to $m$
begin
    for $j$ := 1 to $n$
    begin
        $c_{ij}$ := 0
        for $q$ := 1 to $k$
            $c_{ij}$ := $c_{ij} + a_{iq}b_{qj}$
    end
end {C = $[c_{ij}]$ is the product of A and B}
```

We can determine the complexity of this algorithm in terms of the number of additions and multiplications used.

EXAMPLE 5 How many additions of integers and multiplications of integers are used by Algorithm 1 to multiply two $n \times n$ matrices with integer entries?

Solution: There are n^2 entries in the product of **A** and **B**. To find each entry requires a total of n multiplications and $n - 1$ additions. Hence, a total of n^3 multiplications and $n^2(n - 1)$ additions are used. ■

Surprisingly, there are more efficient algorithms for matrix multiplication than that given in Algorithm 1. As Example 5 shows, multiplying two $n \times n$ matrices directly from the definition requires $O(n^3)$ multiplications and additions. Using other algorithms, two $n \times n$ matrices can be multiplied using $O(n^{\sqrt{7}})$ multiplications and additions. (Details of such algorithms can be found in the references given in the suggested readings at the end of the book.)

There is another important problem involving the complexity of the multiplication of matrices. How should the product $\mathbf{A}_1\mathbf{A}_2 \cdots \mathbf{A}_n$ be computed using the fewest multiplications of integers, where $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n$ are $m_1 \times m_2$, $m_2 \times m_3, \ldots, m_n \times m_{n+1}$ matrices, respectively, and each has integers as entries? (Since matrix multiplication is associative, as shown in Exercise 13 at the end of this section, the order of the multiplication used does not matter.) Before studying this problem, note that $m_1m_2m_3$ multiplications of integers are performed to multiply an $m_1 \times m_2$ matrix and an $m_2 \times m_3$ matrix using Algorithm 1 (see Exercise 23 at the end of this section). The following example illustrates this complexity problem.

EXAMPLE 6 In which order should the matrices $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3$—where $\mathbf{A}_1$ is 30×20, $\mathbf{A}_2$ is 20×40, and $\mathbf{A}_3$ is 40×10, all with integer entries—be multiplied to use the least number of multiplications of integers?

Solution: There are two possible ways to compute $\mathbf{A}_1\mathbf{A}_2\mathbf{A}_3$. These are $\mathbf{A}_1(\mathbf{A}_2\mathbf{A}_3)$ and $(\mathbf{A}_1\mathbf{A}_2)\mathbf{A}_3$.

If $\mathbf{A}_2$ and $\mathbf{A}_3$ are first multiplied, a total of $20 \cdot 40 \cdot 10 = 8000$ multiplications of integers are used to obtain the 20×10 matrix $\mathbf{A}_2\mathbf{A}_3$. Then, to multiply $\mathbf{A}_1$ and $\mathbf{A}_2\mathbf{A}_3$ requires $30 \cdot 20 \cdot 10 = 6000$ multiplications. Hence, a total of

$$8000 + 6000 = 14{,}000$$

multiplications are used. On the other hand, if $\mathbf{A}_1$ and $\mathbf{A}_2$ are first multiplied, then $30 \cdot 20 \cdot 40 = 24{,}000$ multiplications are used to obtain the 30×40 matrix $\mathbf{A}_1\mathbf{A}_2$. Then, to multiply $\mathbf{A}_1\mathbf{A}_2$ and $\mathbf{A}_3$ requires $30 \times 40 \times 10 = 12{,}000$ multiplication. Hence, a total of

$$24{,}000 + 12{,}000 = 36{,}000$$

multiplications are used.

Clearly, the first method is more efficient. ■

Algorithms for determining the most efficient way to multiply n matrices are discussed in the suggested readings listed at the end of the book.

TRANSPOSES AND POWERS OF MATRICES

We now introduce an important matrix with entries that are zeros and ones.

DEFINITION 5. The *identity matrix of order n* is the $n \times n$ matrix $\mathbf{I}_n = [\delta_{ij}]$, where $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ if $i \neq j$. Hence

$$\mathbf{I}_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ 0 & 0 & \cdots & 1 \end{bmatrix}.$$

Multiplying a matrix by an appropriately sized identity matrix does not change this matrix. In other words, when $\mathbf{A}$ is an $m \times n$ matrix, we have

$$\mathbf{A}\mathbf{I}_n = \mathbf{I}_m\mathbf{A} = \mathbf{A}.$$

Powers of square matrices can be defined. When $\mathbf{A}$ is an $n \times n$ matrix, we have

$$\mathbf{A}^0 = \mathbf{I}_n, \qquad \mathbf{A}^r = \underbrace{\mathbf{AAA}\cdots\mathbf{A}}_{r \text{ times}}.$$

The operation of interchanging the rows and columns of a square matrix is used in many algorithms.

DEFINITION 6. Let $\mathbf{A} = [a_{ij}]$ be an $m \times n$ matrix. The *transpose* of $\mathbf{A}$, denoted by $\mathbf{A}^t$, is the $n \times m$ matrix obtained by interchanging the rows and columns of $\mathbf{A}$. In other words, if $\mathbf{A}^t = [b_{ij}]$, then $b_{ij} = a_{ji}$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$.

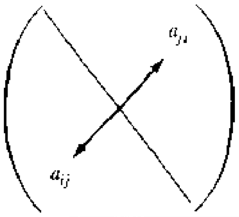


FIGURE 2 A Symmetric Matrix.

EXAMPLE 7 The transpose of the matrix $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$ is the matrix $\begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$. ■

Matrices that do not change when their rows and columns are interchanged are often important.

DEFINITION 7. **A square matrix A is called *symmetric* if $\mathbf{A} = \mathbf{A}^t$. Thus $\mathbf{A} = [a_{ij}]$ is symmetric if $a_{ij} = a_{ji}$ for all i and j with $1 \le i \le n$ and $1 \le j \le n$.**

Note that a matrix is symmetric if and only if it is square and it is symmetric with respect to its main diagonal (which consists of entries that are in the ith row and ith column for some i). This symmetry is displayed in Figure 2.

EXAMPLE 8 The matrix $\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ is symmetric. ■

ZERO–ONE MATRICES

A matrix with entries that are either 0 or 1 is called a **zero–one matrix.** Zero–one matrices are often used to represent discrete structures, as we will see in Chapters 6 and 7. Algorithms using these structures are based on Boolean arithmetic with zero–one matrices. This arithmetic is based on the Boolean operations $\vee$ and $\wedge$, which operate on pairs of bits, defined by

$$b_1 \wedge b_2 = \begin{cases} 1 & \text{if } b_1 = b_2 = 1 \\ 0 & \text{otherwise,} \end{cases}$$

$$b_1 \vee b_2 = \begin{cases} 1 & \text{if } b_1 = 1 \text{ or } b_2 = 1 \\ 0 & \text{otherwise.} \end{cases}$$

DEFINITION 8. **Let $\mathbf{A} = [a_{ij}]$ and $\mathbf{B} = [b_{ij}]$ be $m \times n$ zero–one matrices. Then the *join* of A and B is the zero–one matrix with (i, j)th entry $a_{ij} \vee b_{ij}$. The join of A and B is denoted by $\mathbf{A} \vee \mathbf{B}$. The *meet* of A and B is the zero–one matrix with (i, j)th entry $a_{ij} \wedge b_{ij}$. The meet of A and B is denoted by $\mathbf{A} \wedge \mathbf{B}$.**

EXAMPLE 9 Find the join and meet of the zero–one matrices

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}.$$

Solution: We find that the join of **A** and **B** is

$$\mathbf{A} \vee \mathbf{B} = \begin{bmatrix} 1 \vee 0 & 0 \vee 1 & 1 \vee 0 \\ 0 \vee 1 & 1 \vee 1 & 0 \vee 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

The meet of **A** and **B** is

$$\mathbf{A} \wedge \mathbf{B} = \begin{bmatrix} 1 \wedge 0 & 0 \wedge 1 & 1 \wedge 0 \\ 0 \wedge 1 & 1 \wedge 1 & 0 \wedge 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

■

We now define the **Boolean product** of two matrices.

DEFINITION 9. Let $\mathbf{A} = [a_{ij}]$ be an $m \times k$ zero–one matrix and $\mathbf{B} = [b_{ij}]$ be a $k \times n$ zero–one matrix. Then the *Boolean product* of **A** and **B**, denoted by $\mathbf{A} \odot \mathbf{B}$, is the $m \times n$ matrix with (i, j)th entry $[c_{ij}]$ where

$$c_{ij} = (a_{i1} \wedge b_{1j}) \vee (a_{i2} \wedge b_{2j}) \vee \cdots \vee (a_{ik} \wedge b_{kj}).$$

Note that the Boolean product of **A** and **B** is obtained in an analogous way to the ordinary product of these matrices, but with addition replaced with the operation $\vee$ and with multiplication replaced with the operation $\wedge$. We give an example of the Boolean products of matrices.

EXAMPLE 10 Find the Boolean product of **A** and **B**, where

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

Solution: The Boolean product $\mathbf{A} \odot \mathbf{B}$ is given by

$$\begin{aligned} \mathbf{A} \odot \mathbf{B} &= \begin{bmatrix} (1 \wedge 1) \vee (0 \wedge 0) & (1 \wedge 1) \vee (0 \wedge 1) & (1 \wedge 0) \vee (0 \wedge 1) \\ (0 \wedge 1) \vee (1 \wedge 0) & (0 \wedge 1) \vee (1 \wedge 1) & (0 \wedge 0) \vee (1 \wedge 1) \\ (1 \wedge 1) \vee (0 \wedge 0) & (1 \wedge 1) \vee (0 \wedge 1) & (1 \wedge 0) \vee (0 \wedge 1) \end{bmatrix} \\ &= \begin{bmatrix} 1 \vee 0 & 1 \vee 0 & 0 \vee 0 \\ 0 \vee 0 & 0 \vee 1 & 0 \vee 1 \\ 1 \vee 0 & 1 \vee 0 & 0 \vee 0 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}. \end{aligned}$$

■

Algorithm 2 displays pseudocode for computing the Boolean product of two matrices.

ALGORITHM 2 **The Boolean Product.**

```
procedure Boolean product(A, B: zero–one matrices)
for $i := 1$ to $m$
begin
    for $j := 1$ to $n$
    begin
        $c_{ij} := 0$
        for $q := 1$ to $k$
            $c_{ij} := c_{ij} \vee (a_{iq} \wedge b_{qj})$
    end
end {C = $[c_{ij}]$ is the Boolean product of A and B}
```

We can also define the Boolean powers of a square zero–one matrix. These powers will be used in our subsequent studies of paths in graphs, which are used to model such things as communications paths in computer networks.

DEFINITION 10. **Let A be a square zero–one matrix and let r be a positive integer. The rth *Boolean power* of A is the Boolean product of r factors of A. The rth Boolean product of A is denoted by $\mathbf{A}^{[r]}$. Hence**

$$\mathbf{A}^{[r]} = \underbrace{\mathbf{A} \odot \mathbf{A} \odot \mathbf{A} \odot \cdots \odot \mathbf{A}}_{r \text{ times}}.$$

(This is well defined since the Boolean product of matrices is associative.) We also define $\mathbf{A}^{[0]}$ to be $\mathbf{I}_n$.

EXAMPLE 11 Let $\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$. Find $\mathbf{A}^{[n]}$ for all positive integers n.

Solution: We find that

$$\mathbf{A}^{[2]} = \mathbf{A} \odot \mathbf{A} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}.$$

We also find that

$$\mathbf{A}^{[3]} = \mathbf{A}^{[2]} \odot \mathbf{A} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \qquad \mathbf{A}^{[4]} = \mathbf{A}^{[3]} \odot \mathbf{A} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Additional computation shows that

$$\mathbf{A}^{[5]} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

The reader can now see that $\mathbf{A}^{[n]} = \mathbf{A}^{[5]}$ for all positive integers n with $n \geq 5$. ■

The number of bit operations used to find the Boolean product of two $n \times n$ matrices can be easily determined.

EXAMPLE 12 How many bit operations are used to find $\mathbf{A} \odot \mathbf{B}$, where $\mathbf{A}$ and $\mathbf{B}$ are $n \times n$ zero–one matrices?

Solution: There are n^2 entries in $\mathbf{A} \odot \mathbf{B}$. Using Algorithm 2, a total of n *OR*s and n *AND*s are used to find an entry of $\mathbf{A} \odot \mathbf{B}$. Hence, $2n$ bit operations are used to find each entry. Therefore, $2n^3$ bit operations are required to compute $\mathbf{A} \odot \mathbf{B}$ using Algorithm 2. ■

Exercises

1. Let $\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 & 3 \\ 2 & 0 & 4 & 6 \\ 1 & 1 & 3 & 7 \end{bmatrix}$.

- **a)** What size is $\mathbf{A}$?
- **b)** What is the third column of $\mathbf{A}$?
- **c)** What is the second row of $\mathbf{A}$?
- **d)** What is the element of $\mathbf{A}$ in the (3, 2)th position?
- **e)** What is $\mathbf{A}^t$?

2. Find $\mathbf{A} + \mathbf{B}$, where

a) $\mathbf{A} = \begin{bmatrix} 1 & 0 & 4 \\ -1 & 2 & 2 \\ 0 & -2 & -3 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} -1 & 3 & 5 \\ 2 & 2 & -3 \\ 2 & -3 & 0 \end{bmatrix}$.

b) $\mathbf{A} = \begin{bmatrix} -1 & 0 & 5 & 6 \\ -4 & -3 & 5 & -2 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} -3 & 9 & -3 & 4 \\ 0 & -2 & -1 & 2 \end{bmatrix}$.

3. Find $\mathbf{AB}$ if

a) $\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 3 & 2 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 & 4 \\ 1 & 3 \end{bmatrix}$.

b) $\mathbf{A} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 2 & 3 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 3 & -2 & -1 \\ 1 & 0 & 2 \end{bmatrix}$.

c) $\mathbf{A} = \begin{bmatrix} 4 & -3 \\ 3 & -1 \\ 0 & -2 \\ -1 & 5 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} -1 & 3 & 2 & -2 \\ 0 & -1 & 4 & -3 \end{bmatrix}$.

4. Find the product $\mathbf{AB}$, where

a) $\mathbf{A} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & -1 & -1 \\ -1 & 1 & 0 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 & 1 & -1 \\ 1 & -1 & 0 \\ -1 & 0 & 1 \end{bmatrix}$.

b) $\mathbf{A} = \begin{bmatrix} 1 & -3 & 0 \\ 1 & 2 & 2 \\ 2 & 1 & -1 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 1 & -1 & 2 & 3 \\ -1 & 0 & 3 & -1 \\ -3 & -2 & 0 & 2 \end{bmatrix}$.

c) $\mathbf{A} = \begin{bmatrix} 0 & -1 \\ 7 & 2 \\ -4 & -3 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 4 & -1 & 2 & 3 & 0 \\ -2 & 0 & 3 & 4 & 1 \end{bmatrix}$.

5. Find a matrix $\mathbf{A}$ such that

$$\begin{bmatrix} 2 & 3 \\ 1 & 4 \end{bmatrix} \mathbf{A} = \begin{bmatrix} 3 & 0 \\ 1 & 2 \end{bmatrix}.$$

(*Hint:* Finding $\mathbf{A}$ requires that you solve systems of linear equations.)

6. Find a matrix $\mathbf{A}$ such that

$$\begin{bmatrix} 1 & 3 & 2 \\ 2 & 1 & 1 \\ 4 & 0 & 3 \end{bmatrix} \mathbf{A} = \begin{bmatrix} 7 & 1 & 3 \\ 1 & 0 & 3 \\ -1 & -3 & 7 \end{bmatrix}$$

7. Let $\mathbf{A}$ be an $m \times n$ matrix and let $\mathbf{0}$ be the $m \times n$ matrix that has all entries equal to zero. Show that $\mathbf{A} = \mathbf{0} + \mathbf{A} = \mathbf{A} + \mathbf{0}$.

8. Show that matrix addition is commutative; that is, show that if $\mathbf{A}$ and $\mathbf{B}$ are both $m \times n$ matrices, then $\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$.

9. Show that matrix addition is associative; that is, show that if **A**, **B**, and **C** are all $m \times n$ matrices, then $\mathbf{A} + (\mathbf{B} + \mathbf{C}) = (\mathbf{A} + \mathbf{B}) + \mathbf{C}$.

10. Let **A** be a 3×4 matrix, **B** be a 4×5 matrix, and **C** be a 4×4 matrix. Determine which of the following products are defined and find the size of those that are defined.

a) **AB** **b)** **BA** **c)** **AC**
d) **CA** **e)** **BC** **f)** **CB**

11. What do we know about the sizes of the matrices **A** and **B** if both of the products **AB** and **BA** are defined?

12. In this exercise we show that matrix multiplication is distributive over matrix addition.

- **a)** Suppose that **A** and **B** are $m \times k$ matrices and that **C** is a $k \times n$ matrix. Show that $(\mathbf{A} + \mathbf{B})\mathbf{C} = \mathbf{AC} + \mathbf{BC}$.
- **b)** Suppose that **C** is an $m \times k$ matrix and that **A** and **B** are $k \times n$ matrices. Show that $\mathbf{C}(\mathbf{A} + \mathbf{B}) = \mathbf{CA} + \mathbf{CB}$.

13. In this exercise we show that matrix multiplication is associative. Suppose that **A** is an $m \times p$ matrix, **B** is a $p \times k$ matrix, and **C** is a $k \times n$ matrix. Show that $\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$.

14. The $n \times n$ matrix $\mathbf{A} = [a_{ij}]$ is called a **diagonal matrix** if $a_{ij} = 0$ when $i \neq j$. Show that the product of two $n \times n$ diagonal matrices is again a diagonal matrix. Give a simple rule for determining this product.

15. Let

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

Find a formula for $\mathbf{A}^n$, whenever n is a positive integer.

16. Show that $(\mathbf{A}^t)^t = \mathbf{A}$.

17. Let **A** and **B** be two $n \times n$ matrices. Show that

a) $(\mathbf{A} + \mathbf{B})^t = \mathbf{A}^t + \mathbf{B}^t$. **b)** $(\mathbf{AB})^t = \mathbf{B}^t\mathbf{A}^t$.

If **A** and **B** are $n \times n$ matrices with $\mathbf{AB} = \mathbf{BA} = \mathbf{I}_n$, then **B** is called the **inverse** of **A** (this terminology is appropriate since such a matrix **B** is unique) and **A** is said to be **invertible**. The notation $\mathbf{B} = \mathbf{A}^{-1}$ denotes that **B** is the inverse of **A**.

18. Show that

$$\begin{bmatrix} 2 & 3 & -1 \\ 1 & 2 & 1 \\ -1 & -1 & 3 \end{bmatrix}$$

is the inverse of

$$\begin{bmatrix} 7 & -8 & 5 \\ -4 & 5 & -3 \\ 1 & -1 & 1 \end{bmatrix}.$$

19. Let **A** be a 2×2 matrix with

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

Show that if $ad - bc \neq 0$, then

$$\mathbf{A}^{-1} = \begin{bmatrix} \dfrac{d}{ad - bc} & \dfrac{-b}{ad - bc} \\ \dfrac{-c}{ad - bc} & \dfrac{a}{ad - bc} \end{bmatrix}.$$

20. Let

$$\mathbf{A} = \begin{bmatrix} -1 & 2 \\ 1 & 3 \end{bmatrix}.$$

- **a)** Find $\mathbf{A}^{-1}$. (*Hint:* Use Exercise 19.)
- **b)** Find $\mathbf{A}^3$.
- **c)** Find $(\mathbf{A}^{-1})^3$.
- **d)** Use your answers to (b) and (c) to show that $(\mathbf{A}^{-1})^3$ is the inverse of $\mathbf{A}^3$.

21. Let **A** be an invertible matrix. Show that $(\mathbf{A}^n)^{-1} = (\mathbf{A}^{-1})^n$ whenever n is a positive integer.

22. Let **A** be a matrix. Show that the matrix $\mathbf{AA}^t$ is symmetric. (*Hint:* Show that this matrix equals its transpose with the help of Exercise 17b.)

23. Show that the conventional algorithm uses $m_1m_2m_3$ multiplications to compute the product of the $m_1 \times m_2$ matrix **A** and the $m_2 \times m_3$ matrix **B**.

24. What is the most efficient way to multiply the matrices $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3$ with sizes

- **a)** 20×50, 50×10, 10×40?
- **b)** 10×5, 5×50, 50×1?

25. What is the most efficient way to multiply the matrices $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{A}_3$, and $\mathbf{A}_4$ if the dimensions of these matrices are 10×2, 2×5, 5×20, and 20×3, respectively?

26. **a)** Show that the system of simultaneous linear equations

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned}$$

in the variables $x_1, x_2, \ldots, x_n$ can be expressed as $\mathbf{AX} = \mathbf{B}$, where $\mathbf{A} = [a_{ij}]$, **X** is an $n \times 1$ matrix with x_i the entry in its ith row, and **B** is an $n \times 1$ matrix with b_i the entry in its ith row.

b) Show that if the matrix $\mathbf{A} = [a_{ij}]$ is invertible (as defined in the preamble to Exercise 18), then the solution of the system in part (a) can be found using the equation $\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$.

27. Use Exercises 18 and 26 to solve the system

$$\begin{aligned} 7x_1 - 8x_2 + 5x_3 &= 5 \\ -4x_1 + 5x_2 - 3x_3 &= -3 \\ x_1 - x_2 + x_3 &= 0 \end{aligned}$$

28. Let

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Find

a) $\mathbf{A} \vee \mathbf{B}$. b) $\mathbf{A} \wedge \mathbf{B}$.

c) $\mathbf{A} \odot \mathbf{B}$.

29. Let

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}.$$

Find

a) $\mathbf{A} \vee \mathbf{B}$.

b) $\mathbf{A} \wedge \mathbf{B}$.

c) $\mathbf{A} \odot \mathbf{B}$.

30. Find the Boolean product of **A** and **B**, where

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{bmatrix}.$$

31. Let

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

Find

a) $\mathbf{A}^{[2]}$. b) $\mathbf{A}^{[3]}$. c) $\mathbf{A} \vee \mathbf{A}^{[2]} \vee \mathbf{A}^{[3]}$.

32. Let **A** be a zero–one matrix. Show that

a) $\mathbf{A} \vee \mathbf{A} = \mathbf{A}$.

b) $\mathbf{A} \wedge \mathbf{A} = \mathbf{A}$.

33. In this exercise we show that the meet and join operations are commutative. Let **A** and **B** be $m \times n$ zero–one matrices. Show that

a) $\mathbf{A} \vee \mathbf{B} = \mathbf{B} \vee \mathbf{A}$.

b) $\mathbf{B} \wedge \mathbf{A} = \mathbf{A} \wedge \mathbf{B}$.

34. In this exercise we show that the meet and join operations are associative. Let **A**, **B**, and **C** be $m \times n$ zero–one matrices. Show that

a) $(\mathbf{A} \vee \mathbf{B}) \vee \mathbf{C} = \mathbf{A} \vee (\mathbf{B} \vee \mathbf{C})$.

b) $(\mathbf{A} \wedge \mathbf{B}) \wedge \mathbf{C} = \mathbf{A} \wedge (\mathbf{B} \wedge \mathbf{C})$.

35. We will establish distributive laws of the meet over the join operation in this exercise. Let **A**, **B**, and **C** be $m \times n$ zero–one matrices. Show that

a) $\mathbf{A} \vee (\mathbf{B} \wedge \mathbf{C}) = (\mathbf{A} \vee \mathbf{B}) \wedge (\mathbf{A} \vee \mathbf{C})$.

b) $\mathbf{A} \wedge (\mathbf{B} \vee \mathbf{C}) = (\mathbf{A} \wedge \mathbf{B}) \vee (\mathbf{A} \wedge \mathbf{C})$.

36. Let **A** be an $n \times n$ zero–one matrix. Let **I** be the $n \times n$ identity matrix. Show that $\mathbf{A} \odot \mathbf{I} = \mathbf{I} \odot \mathbf{A} = \mathbf{A}$.

37. In this exercise we will show that the Boolean product of zero–one matrices is associative. Assume that **A** is an $m \times p$ zero–one matrix, **B** is a $p \times k$ zero–one matrix, and **C** is a $k \times n$ zero–one matrix. Show that $\mathbf{A} \odot (\mathbf{B} \odot \mathbf{C}) = (\mathbf{A} \odot \mathbf{B}) \odot \mathbf{C}$.

Key Terms and Results

TERMS

algorithm: a finite set of precise instructions for performing a computation or solving a problem.

searching algorithm: the problem of locating an element in a list

linear search algorithm: a procedure for searching a list element by element

binary search algorithm: a procedure for searching an ordered list by successively splitting the list in half

time complexity: the amount of time required for an algorithm to solve a problem

space complexity: the amount of storage space required for an algorithm to solve a problem

worst-case time complexity: the greatest amount of time required for an algorithm to solve a problem of a given size

average-case time complexity: the average amount of time required for an algorithm to solve a problem of a given size

$a \mid b$ (a divides b): there is an integer c such that $b = ac$

prime: a positive integer greater than 1 with exactly two positive integer divisors

composite: a positive integer greater than 1 that is not prime

Mersenne prime: a prime of the form $2^p - 1$, where p is prime

gcd(a, b) (greatest common divisor of a and b): the largest integer that divides both a and b

relatively prime integers: integers a and b such that $\gcd(a, b) = 1$

pairwise relatively prime integers: a set of integers with the property that every pair of these integers is relatively prime

lcm(a, b) (least common multiple of a and b): the smallest positive integer that is divisible by both a and b

a mod b: the remainder when the integer a is divided by the positive integer b

$a \equiv b \pmod{m}$ (a is congruent to b modulo m): $a - b$ is divisible by m

encryption: the process of making a message secret

decryption: the process of returning a secret message to its original form

$n = (a_k a_{k-1} \cdots a_1 a_0)_b$: the base b representation of n

binary representation: the base 2 representation of an integer

hexadecimal representation: the base 16 representation of an integer

linear combination of a and b with integer coefficients: a number of the form $sa + tb$ where s and t are integers

inverse of a modulo m: an integer $\bar{a}$ such that $\bar{a}a \equiv 1 \pmod{m}$

linear congruence: a congruence of the form $ax \equiv b \pmod{m}$ where x is a variable

pseudoprime to the base 2: a composite integer n such that $2^{n-1} \equiv 1 \pmod{n}$

private key encryption: encryption where both encryption keys and decryption keys must be kept secret

public key encryption: encryption where encryption keys are public knowledge, but decryption keys are kept secret

matrix: a rectangular array of numbers

matrix addition: see page 151

matrix multiplication: see page 152

$\mathbf{I}_n$ (identity matrix of order n): the $n \times n$ matrix that has entries equal to 1 on its diagonal and 0s elsewhere

$\mathbf{A}^t$ (transpose of $\mathbf{A}$): the matrix obtained from $\mathbf{A}$ by interchanging the rows and columns

symmetric: a matrix is symmetric if it equals its transpose

zero–one matrix: a matrix with each entry equal to either 0 or 1

$\mathbf{A} \vee \mathbf{B}$ (the join of $\mathbf{A}$ and $\mathbf{B}$): see page 156

$\mathbf{A} \wedge \mathbf{B}$ (the meet of $\mathbf{A}$ and $\mathbf{B}$): see page 156

$\mathbf{A} \odot \mathbf{B}$ (the Boolean product of $\mathbf{A}$ and $\mathbf{B}$): see page 157

RESULTS

The linear and binary search algorithms: (given in Section 2.1).

The Fundamental Theorem of Arithmetic: Every positive integer can be written uniquely as the product of primes, where the prime factors are written in order of increasing size.

The division algorithm: Let a and d be integers with d positive. Then there are unique integers q and r with $0 \leq r < d$ such that $a = dq + r$.

If a and b are positive integers, then $ab = \gcd(a, b) \cdot \operatorname{lcm}(a, b)$.

The Euclidean algorithm: for finding greatest common divisors (see Algorithm 1 in Section 2.4).

Let b be a positive integer greater than 1. Then if n is a positive integer, it can be expressed uniquely in the form $n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b + a_0$.

The algorithm for finding the base b expansion of an integer (see Algorithm 2 in Section 2.4).

The conventional algorithms for addition and multiplication of integers (given in Section 2.4).

The greatest common divisor of two integers can be expressed as a linear combination with integer coefficients of these integers.

If m is a positive integer and $\gcd(a, m) = 1$, then a has a unique inverse modulo m.

The Chinese Remainder Theorem: A system of linear congruences modulo pairwise relatively prime integers has a unique solution modulo the product of these moduli.

Fermat's Little Theorem: If p is prime and $p \nmid a$, then $a^{p-1} \equiv 1 \pmod{p}$.

Review Questions

1. a) Define the term *algorithm*.
   b) What are the different ways to describe algorithms?
   c) What is the difference between an algorithm for solving a problem and a computer program that solves this problem?
2. a) Describe, using English, an algorithm for finding the largest, second largest, and third largest integers in a list of n integers.
   b) Express this algorithm in pseudocode.
   c) How many comparisons does the algorithm use?
3. a) Define what the worst-case time complexity, average-case time complexity, and best-case time complexity (in terms of comparisons) mean for an algorithm that finds the smallest integer in a list of n integers.
   b) What are the worst-case, average-case, and best-case time complexities, in terms of comparisons, of the algorithm that finds the smallest integer in a list of n integers by comparing each of the integers with the smallest integer found so far?
4. a) Describe the linear search and binary search algorithm for finding an integer in a list of integers in increasing order.
   b) Compare the worst-case time complexities of these two algorithms.
   c) Is one of these algorithms always faster than the other (measured in terms of comparisons)?
5. State the Fundamental Theorem of Arithmetic.
6. a) Describe a procedure for finding the prime factorization of an integer.

b) Use this procedure to find the prime factorization of 80,707.

7. a) Define the greatest common divisor of two integers.

b) Describe at least three different ways to find the greatest common divisor of two integers. When does each method work best?

c) Find the greatest common divisor of 1,234,567 and 7,654,321.

d) Find the greatest common divisor of $2^3 3^5 5^7 7^9 11$ and $2^9 3^7 5^5 7^3 13$.

8. a) Define what it means for a and b to be congruent modulo 7.

b) Which pairs of the integers $-11, -8, -7, -1, 0, 3,$ and 17 are congruent modulo 7?

c) Show that if a and b are congruent modulo 7, then $10a + 13$ and $-4b + 20$ are also congruent modulo 7.

9. Describe a procedure for converting decimal (base 10) expansions of integers into hexadecimal expansions.

10. a) How can you find a linear combination (with integer coefficients) of two integers that equals their greatest common divisor?

b) Express gcd(84, 119) as a linear combination of 84 and 119.

11. a) What does it mean for $\bar{a}$ to be an inverse of a modulo m?

b) How can you find an inverse of a modulo m when m is a positive integer and $\gcd(a, m) = 1$?

c) Find an inverse of 7 modulo 19.

12. a) How can an inverse of a modulo m be used to solve the linear congruence $ax \equiv b \pmod{m}$ when $\gcd(a, m) = 1$?

b) Solve the linear congruence $7x \equiv 13 \pmod{19}$.

13. a) State the Chinese Remainder Theorem.

b) Find the solutions to the system $x \equiv 1 \pmod{4}$, $x \equiv 2 \pmod{5}$, and $x \equiv 3 \pmod{7}$.

14. Suppose that $2^{n-1} \equiv 1 \pmod{n}$. Is n necessarily prime?

15. a) What is the difference between a public key and a private key cryptosystem?

b) Explain why using shift ciphers is a private key system.

c) Explain why the RSA cipher system is a public key system.

16. Define the product of two matrices **A** and **B**. When is this product defined?

17. a) How many different ways are there to evaluate the product $\mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4$ by successively multiplying pairs of matrices, when this product is defined?

b) Suppose that $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{A}_3$, and $\mathbf{A}_4$ are 10×20, 20×5, 5×10, and 10×5 matrices, respectively. How should $\mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4$ be computed to use the least number of multiplications of entries?

Supplementary Exercises

1. a) Describe an algorithm for locating the last occurrence of the largest number in a list of integers.

b) Estimate the number of comparisons used.

2. a) Describe an algorithm for finding the first and second largest elements in a list of integers.

b) Estimate the number of comparisons used.

3. a) Give an algorithm to determine whether a bit string contains a pair of consecutive zeros.

b) How many comparisons does the algorithm use?

4. a) Suppose that a list contains integers that are in order of largest to smallest and an integer can appear repeatedly in this list. Devise an algorithm that locates all occurrences of an integer x in the list.

b) Estimate the number of comparisons used.

5. Find four numbers that are congruent to 5 modulo 17.

6. Show that if a and d are positive integers, then there are integers q and r such that $a = dq + r$ where $-d/2 < r \leq d/2$.

***7.** Show that if $ac \equiv bc \pmod{m}$, then $a \equiv b \pmod{m/d}$ where $d = \gcd(m, c)$.

***8.** How many zeros are at the end of the binary expansion of $100_{10}!$?

9. Use the Euclidean algorithm to find the greatest common divisor of 10,223 and 33,341.

10. How many divisions are required to find gcd(144, 233) using the Euclidean algorithm?

11. Find $\gcd(2n + 1, 3n + 2)$, where n is a positive integer. (*Hint:* Use the Euclidean algorithm.)

12. a) Show that if a and b are positive integers with $a \geq b$, then $\gcd(a, b) = a$ if $a = b$, $\gcd(a, b) = 2\gcd(a/2, b/2)$ if a and b are even, $\gcd(a, b) = \gcd(a/2, b)$ if a is even and b is odd, and $\gcd(a, b) = \gcd(a - b, b)$ if both a and b are odd.

b) Explain how to use (a) to construct an algorithm for computing the greatest common divisor of two positive integers that uses only comparisons, subtractions, and shifts of binary expansions, without using any divisions.

c) Find gcd(1202, 4848) using this algorithm.

13. Show that an integer is divisible by 9 if and only if the sum of its decimal digits is divisible by 9.

14. a) Devise an algorithm for computing x^n **mod** m, where x is an integer and m and n are positive integers, using the binary expansion of n. (*Hint:*

Perform successive squarings to obtain x **mod** m, x^2 **mod** m, x^4 **mod** m, and so on. Then multiply the appropriate powers of the form x^{2^k} **mod** m to obtain x^n **mod** m.)

b) Estimate the number of multiplications used by this algorithm.

A set of integers is called **mutually relatively prime** if the greatest common divisor of these integers is 1.

15. Determine whether the following sets of integers are mutually relatively prime.
  a) 8, 10, 12
  b) 12, 15, 25
  c) 15, 21, 28
  d) 21, 24, 28, 32

16. Find a set of four mutually relatively prime integers such that no two of them are relatively prime.

17. **a)** Suppose that messages are encrypted using the function $f(p) = (ap + b)$ **mod** 26 such that $\gcd(a, 26) = 1$. Determine a function that can be used to decrypt messages.
  b) The encrypted version of a message is LJMKG MGMXF QEXMW. If it was encrypted using the function $f(p) = (7p + 10)$ **mod** 26, what was the original message?

18. Show that the system of congruences $x \equiv 2 \pmod 6$ and $x \equiv 3 \pmod 9$ has no solutions.

19. Find all solutions of the system of congruences $x \equiv 4 \pmod 6$ and $x \equiv 13 \pmod{15}$.

***20.** **a)** Show that the system of congruences $x \equiv a_1 \pmod{m_1}$ and $x \equiv a_2 \pmod{m_2}$ has a solution if and only if $\gcd(m_1, m_2) \mid a_1 - a_2$.
  b) Show that the solution in part (a) is unique modulo $\operatorname{lcm}(m_1, m_2)$.

21. Find $\mathbf{A}^n$ if $\mathbf{A}$ is

$$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

22. Show that if $\mathbf{A} = c\mathbf{I}$, where c is a real number and $\mathbf{I}$ is the $n \times n$ identity matrix, then $\mathbf{AB} = \mathbf{BA}$ whenever $\mathbf{B}$ is an $n \times n$ matrix.

23. Show that if $\mathbf{A}$ is a 2×2 matrix such that $\mathbf{AB} = \mathbf{BA}$ whenever $\mathbf{B}$ is a 2×2 matrix, then $\mathbf{A} = c\mathbf{I}$, where c is a real number and $\mathbf{I}$ is the 2×2 identity matrix.

An $n \times n$ matrix is called **upper triangular** if $a_{ij} = 0$ whenever $i > j$.

24. From the definition of the matrix product, devise an algorithm for computing the product of two upper triangular matrices that ignores those products in the computation that are automatically equal to zero.

25. Give a pseudocode description of the algorithm in Exercise 24 for multiplying two upper triangular matrices.

26. How many multiplications of entries are used by the algorithm found in Exercise 25 for multiplying two $n \times n$ upper triangular matrices?

27. Show that if $\mathbf{A}$ and $\mathbf{B}$ are invertible matrices and $\mathbf{AB}$ exists, then $(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$.

28. What is the best order to form the product $\mathbf{ABCD}$ if $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$ are matrices with dimensions 30×10, 10×40, 40×50, and 50×30, respectively? Assume that the number of multiplications of entries used to multiply a $p \times q$ matrix and a $q \times r$ matrix is pqr.

29. Let $\mathbf{A}$ be an $n \times n$ matrix and let $\mathbf{0}$ be the $n \times n$ matrix all of whose entries are zero. Show that the following are true.
  a) $\mathbf{A} \odot \mathbf{0} = \mathbf{0} \odot \mathbf{A} = \mathbf{0}$ **b)** $\mathbf{A} \vee \mathbf{0} = \mathbf{0} \vee \mathbf{A} = \mathbf{A}$
  c) $\mathbf{A} \wedge \mathbf{0} = \mathbf{0} \wedge \mathbf{A} = \mathbf{0}$

Computer Projects

WRITE PROGRAMS WITH THE FOLLOWING INPUT AND OUTPUT.

1. Given a list of n integers, find the largest integer in the list.

2. Given a list of n integers, find the first and last occurrences of the largest integer in the list.

3. Given a list of n distinct integers, determine the position of an integer in the list using a linear search.

4. Given an ordered list of n distinct integers, determine the position of an integer in the list using a binary search.

5. Given an ordered list of n integers and an integer x, find the number of comparisons used to determine the position of an integer in the list using a linear search and using a binary search.

6. Given a positive integer, determine whether it is prime.

7. Given a message, encrypt this message using the Caesar cipher; and given a message encrypted using the Caesar cipher, decrypt this message.

8. Given two positive integers, find their greatest common divisor using the Euclidean algorithm.

9. Given two positive integers, find their least common multiple.

***10.** Given a positive integer, find the prime factorization of this integer.

11. Given a positive integer and a positive integer b greater than 1, find the base b expansion of this integer.

12. Given a positive integer, find the Cantor expansion of this integer (see the preamble to Exercise 32 of Section 2.4).
13. Given a positive integer n, a modulus m, multiplier a, increment c, and seed x_0, with $0 \leq a < m$, $0 \leq c < m$, and $0 \leq x_0 < m$, generate the sequence of n pseudorandom numbers using the linear congruential generator $x_{n+1} = (ax_n + c)$ **mod** m.
14. Given positive integers a and b, find integers s and t such that $sa + tb = \gcd(a, b)$.
15. Given n linear congruences modulo pairwise relatively prime moduli, find the simultaneous solution of these congruences modulo the product of these moduli.
16. Given an $m \times k$ matrix **A** and a $k \times n$ matrix **B**, find **AB**.
17. Given a square matrix **A** and a positive integer n, find $\mathbf{A}^n$.
18. Given a square matrix, determine whether it is symmetric.
19. Given an $n_1 \times n_2$ matrix **A**, an $n_2 \times n_3$ matrix **B**, an $n_3 \times n_4$ matrix **C**, and an $n_4 \times n_5$ matrix **D**, all with integer entries, determine the most efficient order to multiply these matrices (in terms of the number of multiplications and additions of integers).
20. Given two $m \times n$ Boolean matrices, find their meet and join.
21. Given an $m \times k$ Boolean matrix **A** and a $k \times n$ Boolean matrix **B**, find the Boolean product of **A** and **B**.
22. Given a square Boolean matrix **A** and a positive integer n, find $\mathbf{A}^{[n]}$.

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. Determine whether $2^p - 1$ is prime for each of the primes not exceeding 100.
2. Test a range of large Mersenne numbers $2^p - 1$ to determine whether they are prime. (You may want to use software from the GIMPS project.)
3. Show that $n^2 + n + 41$ is prime for all integers n with $0 \leq n \leq 39$, but is not prime when $n = 40$. Is there a polynomial in n with integer coefficients and degree greater than zero that always takes on a prime value when n is a positive integer?
4. Find as many primes of the form $n^2 + 1$ where n is a positive integer as you can. It is not known whether there are infinitely many such primes.
5. Find 10 different primes each with 100 digits.
6. How many primes are there less than 1,000,000, less than 10,000,000, and less than 100,000,000? Can you propose an estimate for the number of primes less than x where x is a positive integer?
7. Find a prime factor of each of 10 different 20-digit odd integers, selected at random. Keep track of how long it takes to find a factor of each of these integers. Do the same thing for 10 different 30-digit odd integers, 10 different 40-digit odd integers, and so on, continuing as long as possible.
8. Find all pseudoprimes to the base 2, that is, composite integers n such that $2^{n-1} \equiv 1 \pmod{n}$, where n does not exceed 10,000.

Writing Projects

RESPOND TO THE FOLLOWING PROJECTS WITH ESSAYS USING OUTSIDE SOURCES.

1. Examine the history of the word *algorithm* and describe the use of this word in early writings.
2. Describe what is meant by a parallel algorithm. Explain how the pseudocode used in this book can be extended to handle parallel algorithms.
3. Explain how the complexity of parallel algorithms can be measured. Give some examples to illustrate this concept, showing how a parallel algorithm can work more quickly than one that does not operate in parallel.
4. Describe the Lucas–Lehmer test for determining whether a Mersenne number is prime. Discuss the progress of the GIMPS project in finding Mersenne primes using this test.
5. Explain how probabilistic primality tests are used in practice to produce extremely large numbers that are almost certainly prime. Do such tests have any potential drawbacks?
6. A *Carmichael number* is an integer that is a pseudoprime to all bases relatively prime to this integer. The question of whether there are infinitely many Carmichael numbers was solved recently after being open for more than 75 years. Explain what a

Carmichael number is, give examples of such numbers, and describe the ingredients that went into the proof that there are infinitely many such numbers.

7. Summarize the current status of factoring algorithms in terms of their complexity and the size of numbers that can currently be factored. When do you think that it will be feasible to factor 200-digit numbers?
8. Describe the algorithms that are actually used in modern computers to add, subtract, multiply, and divide positive integers.
9. Describe the history of the Chinese Remainder Theorem. Describe some of the relevant problems posed in Chinese and Hindu writings and how the Chinese Remainder Theorem applies to them.
10. When are the numbers of a sequence truly random numbers, and not pseudorandom? What shortcomings have been observed in simulations and experiments in which pseudorandom numbers have been used? What are the properties that pseudorandom numbers can have that random numbers should not have?
11. Describe how public key cryptography is being applied. Are the ways it is applied secure given the status of factoring algorithms? Will information kept secure using public key cryptography become insecure in the future?
12. Describe how public key cryptography can be used to send signed secret messages so that the recipient is relatively sure the message was sent by the person claiming to have sent it.
13. Show how a congruence can be used to tell the day of the week for any given date.
14. Describe some of the algorithms used to efficiently multiply large integers.
15. Describe some of the algorithms used to efficiently multiply large matrices.

Mathematical Reasoning

3

To understand written mathematics, we must understand what makes up a correct mathematical argument, that is, a proof. To learn mathematics, a person needs to construct mathematical arguments and not just read exposition. Obviously, this requires an understanding of the techniques used to build proofs. The goals of this chapter are to teach what makes up a correct mathematical argument and to give the student the necessary tools to construct these arguments.

Note that the methods we will study for building proofs are also used throughout computer science, such as the rules computers use to reason, the techniques used to verify that programs are correct, and the rules used for constructing new theorems using automated reasoning.

Many mathematical statements assert that a property is true for all positive integers. Examples of such statements are that for every positive integer n: $n! \leq n^n$, $n^3 - n$ is divisible by 3, and the sum of the first n positive integers is $n(n + 1)/2$. A major goal of this chapter, and the book, is to give the student a thorough understanding of mathematical induction, which is used to prove results of this kind.

In previous chapters we explicitly defined sets, sequences, and functions. That is, we described sets by listing their elements or by giving some property that characterizes these elements. We gave formulae for the terms of sequences and the values of functions. There is another important way to define such objects, based on mathematical induction. To define sequences and functions, some initial terms are specified, and a rule is given for finding subsequent values from values already known. For instance, we can define the sequence $\{2^n\}$ by specifying that $a_1 = 2$ and that $a_{n+1} = 2a_n$ for $n = 1, 2, 3, \ldots$. Sets can be defined by listing some of their elements and giving rules for constructing elements from those already known to be in the set. Such definitions, called *recursive definitions*, are used throughout discrete mathematics and computer science.

When a procedure is specified for solving a problem, this procedure *always* solves the problem correctly. Just testing to see that the correct result is obtained for a set of input values does not show that the procedure always works correctly. The correctness of a procedure can be guaranteed only by proving that it always yields the correct result. The final section of this chapter contains an introduction to the techniques of program verification. This is a formal technique to verify that procedures are correct. Program verification serves as the basis for attempts under way to prove in a mechanical fashion that programs are correct.

3.1

Methods of Proof

INTRODUCTION

Two important questions that arise in the study of mathematics are: (1) When is a mathematical argument correct? (2) What methods can be used to construct mathematical arguments? This section helps answer these questions by describing various forms of correct and incorrect mathematical arguments.

A **theorem** is a statement that can be shown to be true. We demonstrate that a theorem is true with a sequence of statements that form an argument, called a **proof.** To construct proofs, methods are needed to derive new statements from old ones. The statements used in a proof can include **axioms** or **postulates,** which are the underlying assumptions about mathematical structures, the hypotheses of the theorem to be proved, and previously proved theorems. The **rules of inference,** which are the means used to draw conclusions from other assertions, tie together the steps of a proof.

In this section rules of inference will be discussed. This will help clarify what makes up a correct proof. Some common forms of incorrect reasoning, called **fallacies,** will also be described. Then various methods commonly used to prove theorems will be introduced.

Remark: The terms *lemma* and *corollary* are used for certain types of theorems. A **lemma** (plural **lemmas** or **lemmata**) is a simple theorem used in the proof of other theorems. (For instance, Lemma 1 in Section 2.4 was used to prove the theorem that the Euclidean algorithm produces the greatest common divisor of two integers.) Complicated proofs are usually easier to understand when they are proved using a series of lemmas, where each lemma is proved individually. A **corollary** is a proposition that can be established directly from a theorem that has been proved. A **conjecture** is a statement whose truth value is unknown. When a proof of a conjecture is found, the conjecture becomes a theorem. Many times conjectures are shown to be false, so they are not theorems.

The methods of proof discussed in this chapter are important not only because they are used to prove mathematical theorems, but also for their many applications to computer science. These applications include verifying that computer programs are correct, establishing that operating systems are secure, making inferences in the area of artificial intelligence, and so on. Consequently, understanding the techniques used in proofs is essential both in mathematics and in computer science.

RULES OF INFERENCE

We will now introduce rules of inference for propositional logic. These rules provide the justification of the steps used to show that a conclusion follows logically from a set of hypotheses. The tautology $(p \wedge (p \rightarrow q)) \rightarrow q$ is the basis of the rule of inference called **modus ponens,** or the **law of detachment.** This tautology is written in the following way:

$$\begin{array}{l} p \\ p \rightarrow q \\ \hline \therefore q \end{array}$$

Using this notation, the hypotheses are written in a column and the conclusion below a bar. (The symbol ∴ denotes "therefore.") Modus ponens states that if both an implication and its hypothesis are known to be true, then the conclusion of this implication is true.

EXAMPLE 1 Suppose that the implication "if it snows today, then we will go skiing" and its hypothesis, "it is snowing today," are true. Then, by modus ponens, it follows that the conclusion of the implication, "we will go skiing," is true. ■

TABLE 1 Rules of Inference.

Rule of Inference	*Tautology*	*Name*
p $\therefore p \vee q$	$p \to (p \vee q)$	Addition
$\begin{array}{l} p \wedge q \\ \hline \therefore p \end{array}$	$(p \wedge q) \to p$	Simplification
$\begin{array}{l} p \\ q \\ \hline \therefore p \wedge q \end{array}$	$((p) \wedge (q)) \to (p \wedge q)$	Conjuction
$\begin{array}{l} p \\ p \to q \\ \hline \therefore q \end{array}$	$[p \wedge (p \to q)] \to q$	Modus ponens
$\begin{array}{l} \neg q \\ p \to q \\ \hline \therefore \neg p \end{array}$	$[\neg q \wedge (p \to q)] \to \neg p$	Modus tollens
$\begin{array}{l} p \to q \\ q \to r \\ \hline \therefore p \to r \end{array}$	$[(p \to q) \wedge (q \to r)] \to (p \to r)$	Hypothetical syllogism
$\begin{array}{l} p \vee q \\ \neg p \\ \hline \therefore q \end{array}$	$[(p \vee q) \wedge \neg p] \to q$	Disjunctive syllogism

EXAMPLE 2 The implication "if n is divisible by 3, then n^2 is divisible by 9," is true. Consequently, if n is divisible by 3, then by modus ponens, it follows that n^2 is divisible by 9. ■

Table 1 lists some important rules of inference. The verifications of these rules of inference can be found as exercises in Section 1.2. Here are some examples of arguments using these rules of inference.

EXAMPLE 3 State which rule of inference is the basis of the following argument: "It is below freezing now. Therefore, it is either below freezing or raining now."

Solution: Let p be the proposition "It is below freezing now" and q the proposition "It is raining now." Then this argument is of the form

$$\begin{array}{l} p \\ \hline \therefore p \vee q \end{array}$$

This is an argument that uses the addition rule. ■

EXAMPLE 4 State which rule of inference is the basis of the following argument: "It is below freezing and raining now. Therefore, it is below freezing now."

Solution: Let p be the proposition "It is below freezing now," and let q be the proposition "It is raining now." This argument is of the form

$$\begin{array}{l} p \wedge q \\ \hline \therefore p \end{array}$$

This argument uses the simplification rule. ■

EXAMPLE 5 State which rule of inference is used in the argument:

> If it rains today, then we will not have a barbecue today. If we do not have a barbecue today, then we will have a barbecue tomorrow. Therefore, if it rains today, then we will have a barbecue tomorrow.

Solution: Let p be the proposition "It is raining today," let q be the proposition "We will not have a barbecue today," and let r be the proposition "We will have a barbecue tomorrow." Then this argument is of the form

$$\begin{array}{l} p \rightarrow q \\ q \rightarrow r \\ \hline \therefore p \rightarrow r \end{array}$$

Hence, this argument is a hypothetical syllogism. ■

An argument is called **valid** if whenever all the hypotheses are true, the conclusion is also true. Consequently, showing that q logically follows from the hypotheses $p_1, p_2, \ldots, p_n$ is the same as showing that the implication

$$(p_1 \wedge p_2 \wedge \cdots \wedge p_n) \rightarrow q$$

is true. When all propositions used in a valid argument are true, it leads to a correct conclusion. However, a valid argument can lead to an incorrect conclusion if one or more false propositions are used within the argument. For example,

> "If 101 is divisible by 3, then 101^2 is divisible by 9. 101 is divisible by 3. Consequently, 101^2 is divisible by 9."

is a valid argument based on modus ponens. However, the conclusion of this argument is false, since 9 does not divide $101^2 = 10{,}201$. The false proposition "101 is divisible by 3" has been used in the argument, which means that the conclusion of the argument may be false.

When there are many premises, several rules of inference are often needed to show that an argument is valid. This is illustrated by the following examples, where the steps of arguments are displayed step by step, with the reason for each step explicitly stated. These examples also show how arguments in English can be analyzed using rules of inference.

EXAMPLE 6 Show that the hypotheses "It is not sunny this afternoon and it is colder than yesterday," "We will go swimming only if it is sunny," "If we do not go swimming, then we will take a canoe trip," and "If we take a canoe trip, then we will be home by sunset" lead to the conclusion "We will be home by sunset."

Solution: Let p be the proposition "It is sunny this afternoon," q the proposition "It is colder than yesterday," r the proposition "We will go swimming," s the proposition "We will take a canoe trip," and t the proposition "We will be home by sunset." Then the hypotheses become $\neg p \wedge q$, $r \rightarrow p$, $\neg r \rightarrow s$, and $s \rightarrow t$. The conclusion is simply t.

We construct an argument to show that our hypotheses lead to the desired conclusion as follows.

Step	**Reason**
1. $\neg p \wedge q$	Hypothesis
2. $\neg p$	Simplification using Step 1
3. $r \rightarrow p$	Hypothesis
4. $\neg r$	Modus tollens using Steps 2 and 3
5. $\neg \rightarrow r$	Hypothesis
6. s	Modus ponens using Steps 4 and 5
7. $s \rightarrow t$	Hypothesis
8. t	Modus ponens using Steps 6 and 7 ■

EXAMPLE 7 Show that the hypotheses "If you send me an e-mail message, then I will finish writing the program," "If you do not send me an e-mail message, then I will go to sleep early," and "If I go to sleep early, then I will wake up feeling refreshed" lead to the conclusion "If I do not finish writing the program, then I will wake up feeling refreshed."

Solution: Let p be the proposition "You send me an e-mail message," q the proposition "I will finish writing the program," r the proposition "I will go to sleep early," and s the proposition "I will wake up feeling refreshed." Then the hypotheses are $p \rightarrow q$, $\neg p \rightarrow r$, and $r \rightarrow s$. The desired conclusion is $\neg q \rightarrow s$.

The following argument shows that our hypotheses lead to the desired conclusion.

Step	**Reason**
1. $p \rightarrow q$	Hypothesis
2. $\neg q \rightarrow \neg p$	Contrapositive of Step 1
3. $\neg p \rightarrow r$	Hypothesis
4. $\neg q \rightarrow r$	Hypothetical syllogism using Steps 2 and 3
5. $r \rightarrow s$	Hypothesis
6. $\neg q \rightarrow s$	Hypothetical syllogism using Steps 4 and 5 ■

FALLACIES

Several common fallacies arise in incorrect arguments. These fallacies resemble rules of inference but are based on contingencies rather than tautologies. These are discussed here to show the distinction between correct and incorrect reasoning.

The proposition $[(p \rightarrow q) \wedge q] \rightarrow p$ is not a tautology, since it is false when p is false and q is true. However, there are many incorrect arguments that treat this as a tautology. This type of incorrect reasoning is called the **fallacy of affirming the conclusion.**

EXAMPLE 8 Is the following argument valid?

> If you do every problem in this book, then you will learn discrete mathematics. You learned discrete mathematics.
>
> Therefore, you did every problem in this book.

Solution: Let p be the proposition "You did every problem in this book." Let q be the proposition "You learned discrete mathematics." Then this argument is of the form: if $p \rightarrow q$ and q, then p. This is an example of an incorrect argument using the fallacy of affirming the conclusion. Indeed, it is possible for you to learn discrete mathematics in some way other than by doing every problem in this book. (You may learn discrete mathematics by reading, listening to lectures, doing some but not all the problems in this book, and so on.) ■

EXAMPLE 9 Let p be the proposition "$n \equiv 1 \pmod{3}$," and let q be the proposition "$n^2 \equiv 1 \pmod{3}$." The implication $p \rightarrow q$, which is "if $n \equiv 1 \pmod{3}$, then $n^2 \equiv 1 \pmod{3}$," is true. If q is true, so that $n^2 \equiv 1 \pmod{3}$, does it follow that p is true, namely, that $n \equiv 1 \pmod{3}$?

Solution: It would be incorrect to conclude that p is true, since it is possible that $n \equiv 2 \pmod{3}$. If the incorrect conclusion that p is true is made, this would be an example of the fallacy of affirming the conclusion. ■

The proposition $[(p \rightarrow q) \wedge \neg p] \rightarrow \neg q$ is not a tautology, since it is false when p is false and q is true. Many incorrect arguments use this incorrectly as a rule of inference. This type of incorrect reasoning is called the **fallacy of denying the hypothesis.**

EXAMPLE 10 Let p and q be as in Example 8. If the implication $p \rightarrow q$ is true, and $\neg p$ is true, is it correct to conclude that $\neg q$ is true? In other words, is it correct to assume that you did not learn discrete mathematics if you did not do every problem in the book, assuming that if you do every problem in this book, then you will learn discrete mathematics?

Solution: It is possible that you learned discrete mathematics even if you did not do every problem in this book. This incorrect argument is of the form $p \rightarrow q$ and $\neg p$ imply $\neg q$, which is an example of the fallacy of denying the hypothesis. ■

EXAMPLE 11 Let p and q be as in Example 9. Is it correct to assume that if $\neg p$ is true, then $\neg q$ is true, using the fact that $p \rightarrow q$ is true? In other words, is it correct to conclude that $n^2 \not\equiv 1 \pmod{3}$ if $n \not\equiv 1 \pmod{3}$, using the implication: if $n \equiv 1 \pmod{3}$, then $n^2 \equiv 1 \pmod{3}$?

Solution: It is incorrect to conclude that $n^2 \not\equiv 1 \pmod 3$ if $n \not\equiv 1 \pmod 3$, since $n^2 \equiv 1 \pmod 3$ when $n \equiv 2 \pmod 3$. This incorrect argument is another example of the fallacy of denying the hypothesis. ■

Many incorrect arguments are based on a fallacy called **begging the question.** This fallacy occurs when one or more steps of a proof are based on the truth of the statement being proved. In other words, this fallacy arises when a statement is proved using itself, or a statement equivalent to it. That is why this fallacy is also called **circular reasoning.**

EXAMPLE 12 Is the following argument correct? It supposedly shows that n is an even integer whenever n^2 is an even integer.

> Suppose that n^2 is even. Then $n^2 = 2k$ for some integer k. Let $n = 2l$ for some integer l. This shows that n is even.

Solution: This argument is incorrect. The statement "let $n = 2l$ for some integer l" occurs in the proof. No argument has been given to show that it is true. This is circular reasoning because this statement is equivalent to the statement being proved, namely, "n is even." Of course, the result itself is correct; only the method of proof is wrong. ■

RULES OF INFERENCE FOR QUANTIFIED STATEMENTS

We discussed rules of inference for propositions. We will now describe some important rules of inference for statements involving quantifiers. These rules of inference are used extensively in mathematical arguments, often without being explicitly mentioned.

Universal instantiation is the rule of inference used to conclude that $P(c)$ is true, where c is a particular member of the universe of discourse, given the premise $\forall x P(x)$. Universal instantiation is used when we conclude from the statement "All women are wise" that "Lisa is wise," where Lisa is a member of the universe of discourse of all women.

Universal generalization is the rule of inference which states that $\forall x P(x)$ is true, given the premise that $P(c)$ is true for all elements c in the universe of discourse. Universal generalization is used when we show that $\forall x P(x)$ is true by taking an arbitrary element c from the universe of discourse and showing that $P(c)$ is true. The element c that we select must be an arbitrary, and not a specific, element of the universe of discourse. Universal generalization is used implicitly in many proofs in mathematics and is seldom mentioned explicitly.

Existential instantiation is the rule which allows us to conclude that there is an element c in the universe of discourse for which $P(c)$ is true if we know that $\exists x P(x)$ is true. We cannot select an arbitrary value of c here, but rather it must be a c for which $P(c)$ is true. Usually we have no knowledge of what c is, only that it exists. Since it exists, we may give it a name (c) and continue our argument.

Existential generalization is the rule of inference which is used to conclude that $\exists x P(x)$ is true when a particular element c with $P(c)$ true is known. That is, if we know one element c in the universe of discourse for which $P(c)$ is true, then we know that $\exists x P(x)$ is true.

We summarize these rules of inference in Table 2.

We will illustrate how one of these rules of inference for quantified statements is used in the following example.

TABLE 2 Rules of Inference for Quantified Statements. *U* Is the Universe of Discourse.

Rule of Inference	*Name*
$\dfrac{\forall x P(x)}{\therefore P(c) \text{ if } c \in U}$	Universal instantiation
$\dfrac{P(c) \text{ for an arbitrary } c \in U}{\therefore \forall x P(x)}$	Universal generalization
$\dfrac{\exists x P(x)}{\therefore P(c) \text{ for some element } c \in U}$	Existential instantiation
$\dfrac{P(c) \text{ for some element } c \in U}{\therefore \exists x P(x)}$	Existential generalization

EXAMPLE 13 Show that the premises "Everyone in this discrete mathematics class has taken a course in computer science" and "Marla is a student in this class" imply the conclusion "Marla has taken a course in computer science."

Solution: Let $D(x)$ denote "x is in this discrete mathematics class," and let $C(x)$ denote "x has taken a course in computer science." Then the premises are $\forall x(D(x) \rightarrow C(x))$ and $D(\text{Marla})$. The conclusion is $C(\text{Marla})$.

The following steps can be used to establish the conclusion from the premises.

Step	Reason
1. $\forall x(D(x) \rightarrow C(x))$	Premise
2. $D(\text{Marla}) \rightarrow C(\text{Marla})$	Universal instantiation using Step 1
3. $D(\text{Marla})$	Premise
4. $C(\text{Marla})$	Modus ponens using Steps 2 and 3 ■

Remark: Mathematical arguments often include steps where both a rule of inference for propositions and a rule of inference for quantifiers are used. For example, universal instantiation and modus ponens are often used together. When these rules of inference are combined, the hypothesis $\forall x(P(x) \rightarrow Q(x))$ and $P(c)$, where c is a member of the universe of discourse, show that the conclusion $Q(c)$ is true.

Remark: Many theorems in mathematics state that a property holds for all elements in a particular set, such as the set of integers or the set of real numbers. Although the precise statement of such theorems needs to include a universal quantifier, the standard convention in mathematics is to omit it. For example, the statement "If the integer n is divisible by 3, then n^2 is divisible by 9" really means "For all integers n, if n is divisible by 3, then n^2 is divisible by 9." Similarly, the statement "If $x > y$, where x and y are positive real numbers, then $x^2 > y^2$" really means "For all positive real numbers x and y, if $x > y$, then $x^2 > y^2$." Furthermore, when theorems of this type are proved, the law of universal generalization is often used without explicit mention.

The first step of the proof usually involves selecting a general element of the universe of discourse. Subsequent steps show that this element has the property in question. Universal generalization implies that the theorem holds for all members of the universe of discourse.

In our subsequent discussions, we will follow the usual conventions and not explicitly mention the use of universal quantification and universal generalization. However, you should always understand when this rule of inference is being implicitly applied.

METHODS OF PROVING THEOREMS

We proved several theorems in Chapters 1 and 2. Let us now be more explicit about the methodology of constructing proofs. We will describe how different types of statements are proved.

Because many theorems are implications, the techniques for proving implications are important. Recall that $p \rightarrow q$ is true unless p is true but q is false. Note that when the statement $p \rightarrow q$ is proved, it need only be shown that q is true if p is true; it is *not* usually the case that q is proved to be true. The following discussion will give the most common techniques for proving implications.

The implication $p \rightarrow q$ can be proved by showing that if p is true, then q must also be true. This shows that the combination p true and q false never occurs. A proof of this kind is called a **direct proof.** To carry out such a proof, assume that p is true and use rules of inference and theorems already proved to show that q must also be true.

EXAMPLE 14 Give a direct proof of the theorem "If n is odd, then n^2 is odd."

Solution: Assume that the hypothesis of this implication is true, namely, suppose that n is odd. Then $n = 2k + 1$, where k is an integer. It follows that $n^2 = (2k + 1)^2 = 4k^2 + 4k + 1 = 2(2k^2 + 2k) + 1$. Therefore, n^2 is odd (it is 1 more than twice an integer). ■

Since the implication $p \rightarrow q$ is equivalent to its contrapositive, $\neg q \rightarrow \neg p$, the implication $p \rightarrow q$ can be proved by showing that its contrapositive, $\neg q \rightarrow \neg p$, is true. This related implication is usually proved directly, but any proof technique can be used. An argument of this type is called an **indirect proof.**

EXAMPLE 15 Give an indirect proof of the theorem "If $3n + 2$ is odd, then n is odd."

Solution: Assume that the conclusion of this implication is false; namely, assume that n is even. Then $n = 2k$ for some integer k. It follows that $3n + 2 = 3(2k) + 2 = 6k + 2 = 2(3k + 1)$, so $3n + 2$ is even (since it is a multiple of 2). Since the negation of the conclusion of the implication implies that the hypothesis is false, the original implication is true. ■

Suppose that the hypothesis p of an implication $p \rightarrow q$ is false. Then the implication $p \rightarrow q$ is true, because the statement has the form F → T or F → F, and hence is true. Consequently, if it can be shown that p is false, then a proof, called a **vacuous proof,** of the implication $p \rightarrow q$ can be given. Vacuous proofs are often used to establish special cases of theorems that state that an implication is true for all positive integers [i.e., a theorem of the kind $\forall n\ P(n)$ where $P(n)$ is a propositional function]. Proof techniques for theorems of this kind will be discussed in Section 3.2.

EXAMPLE 16 Show that the proposition $P(0)$ is true where $P(n)$ is the propositional function "If $n > 1$, then $n^2 > n$."

Solution: Note that the proposition $P(0)$ is the implication "If $0 > 1$, then $0^2 > 0$." Since the hypothesis $0 > 1$ is false, the implication $P(0)$ is automatically true. ■

Remark: The fact that the conclusion of this implication, $0^2 > 0$, is false is irrelevant to the truth value of the implication, because an implication with a false hypothesis is guaranteed to be true.

Suppose that the conclusion q of an implication $p \rightarrow q$ is true. Then $p \rightarrow q$ is true, since the statement has the form $\mathbf{T} \rightarrow \mathbf{T}$ or $\mathbf{F} \rightarrow \mathbf{T}$, which are true. Hence, if it can be shown that q is true, then a proof, called a **trivial proof,** of $p \rightarrow q$ can be given. Trivial proofs are often important when special cases of theorems are proved (see the discussion of proof by cases) and in mathematical induction, which is a proof technique discussed in Section 3.2.

EXAMPLE 17 Let $P(n)$ be the proposition "If a and b are positive integers with $a \geq b$, then $a^n \geq b^n$." Show that the proposition $P(0)$ is true.

Solution: The proposition $P(0)$ is "If $a \geq b$, then $a^0 \geq b^0$." Since $a^0 = b^0 = 1$, the conclusion of $P(0)$ is true. Hence, $P(0)$ is true. This is an example of a trivial proof. Note that the hypothesis, which is the statement "$a \geq b$," was not needed in this proof. ■

Suppose that a contradiction q can be found so that $\neg p \rightarrow q$ is true, that is, $\neg p \rightarrow \mathbf{F}$ is true. Then the proposition $\neg p$ must be false. Consequently, p must be true. This technique can be used when a contradiction, such as $r \wedge \neg r$, can be found so that it is possible to show that the implication $\neg p \rightarrow (r \wedge \neg r)$ is true. An argument of this type is called a **proof by contradiction.**

EXAMPLE 18 Prove that $\sqrt{2}$ is irrational by giving a proof by contradiction.

Solution: Let p be the proposition "$\sqrt{2}$ is irrational." Suppose that $\neg p$ is true. Then $\sqrt{2}$ is rational. We will show that this leads to a contradiction. Under the assumption that $\sqrt{2}$ is rational, there exist integers a and b with $\sqrt{2} = a/b$, where a and b have no common factors (so that the fraction a/b is in lowest terms). Since $\sqrt{2} = a/b$, when both sides of this equation are squared, it follows that

$$2 = a^2/b^2.$$

Hence,

$$2b^2 = a^2.$$

This means that a^2 is even, implying that a is even. Furthermore, since a is even, $a = 2c$ for some integer c. Thus

$$2b^2 = 4c^2,$$

so

$$b^2 = 2c^2.$$

This means that b^2 is even. Hence, b must be even as well.

It has been shown that $\neg p$ implies that $\sqrt{2} = a/b$, where a and b have no common factors, and 2 divides a and b. This is a contradiction since we have shown that $\neg p$ implies both r and $\neg r$ where r is the statement that a and b are integers with no common factors. Hence, $\neg p$ is false, so that p: "$\sqrt{2}$ is irrational" is true. ■

An indirect proof of an implication can be rewritten as a proof by contradiction. In an indirect proof we show that $p \to q$ is true by using a direct proof to show that $\neg q \to \neg p$ is true. That is, in an indirect proof of $p \to q$ we assume that $\neg q$ is true and show that $\neg p$ must also be true. To rewrite an indirect proof of $p \to q$ as a proof by contradiction, we suppose that both p and $\neg q$ are true. Then we use the steps from the direct proof of $\neg q \to \neg p$ to show that $\neg p$ must also be true. This leads to the contradiction $p \wedge \neg p$, completing the proof by contradiction. Example 19 illustrates how an indirect proof of an implication can be rewritten as a proof by contradiction.

EXAMPLE 19 Give a proof by contradiction of the theorem "If $3n + 2$ is odd, then n is odd."

Solution: We assume that $3n+2$ is odd and that n is not odd, so that n is even. Following the same steps as in the solution of Example 15 (an indirect proof of this theorem), we can show that if n is even, then $3n + 2$ is even. This contradicts the assumption that $3n + 2$ is odd, completing the proof. ■

To prove an implication of the form

$$(p_1 \vee p_2 \vee \cdots \vee p_n) \to q$$

the tautology

$$[(p_1 \vee p_2 \vee \cdots \vee p_n) \to q] \leftrightarrow [(p_1 \to q) \wedge (p_2 \to q) \wedge \cdots \wedge (p_n \to q)]$$

can be used as a rule of inference. This shows that the original implication with a hypothesis made up of a disjunction of the propositions $p_1, p_2, \ldots, p_n$ can be proved by proving each of the n implications $p_i \to q$, $i = 1, 2, \ldots, n$, individually. Such an argument is called a **proof by cases.** Sometimes to prove that an implication $p \to q$ is true, it is convenient to use a disjunction $p_1 \vee p_2 \vee \cdots \vee p_n$ instead of p as the hypothesis of the implication, where p and $p_1 \vee p_2 \vee \cdots \vee p_n$ are equivalent. Consider the following example.

EXAMPLE 20 Prove the implication "If n is an integer not divisible by 3, then $n^2 \equiv 1 \pmod{3}$."

Solution: Let p be the proposition "n is not divisible by 3," and let q be the proposition "$n^2 \equiv 1 \pmod{3}$." Then p is equivalent to $p_1 \vee p_2$ where p_1 is "$n \equiv 1 \pmod{3}$" and p_2 is "$n \equiv 2 \pmod{3}$." Hence, to show that $p \rightarrow q$ it can be shown that $p_1 \rightarrow q$ and $p_2 \rightarrow q$. It is easy to give direct proofs of these two implications.

First, suppose that p_1 is true. Then $n \equiv 1 \pmod{3}$, so that $n = 3k + 1$ for some integer k. Thus,

$$n^2 = 9k^2 + 6k + 1 = 3(3k^2 + 2k) + 1.$$

It follows that $n^2 \equiv 1 \pmod{3}$. Hence, the implication $p_1 \rightarrow q$ is true. Next, suppose that p_2 is true. Then $n \equiv 2 \pmod{3}$, so that $n = 3k + 2$ for some integer k. Thus,

$$n^2 = 9k^2 + 12k + 4 = 3(3k^2 + 4k + 1) + 1.$$

Hence, $n^2 \equiv 1 \pmod{3}$, so the implication $p_2 \rightarrow q$ is true.

Since it has been shown that both $p_1 \rightarrow q$ and $p_2 \rightarrow q$ are true, it can be concluded that $(p_1 \vee p_2) \rightarrow q$ is true. Moreover, since p is equivalent to $p_1 \vee p_2$, it follows that $p \rightarrow q$ is true. ■

To prove a theorem that is an equivalence, that is, one that is a statement of the form $p \leftrightarrow q$ where p and q are propositions, the tautology

$$(p \leftrightarrow q) \leftrightarrow [(p \rightarrow q) \wedge (q \rightarrow p)]$$

can be used. That is, the proposition "p if and only if q" can be proved if both the implications "if p, then q" and "if q, then p" are proved.

EXAMPLE 21 *Prove the theorem "The integer n is odd if and only if n^2 is odd."*

Solution: This theorem has the form "p if and only if q," where p is "n is odd" and q is "n^2 is odd." To prove this theorem, we need to show that $p \rightarrow q$ and $q \rightarrow p$ are true.

We have already shown (in Example 14) that $p \rightarrow q$ is true. We will use an indirect proof to prove that $q \rightarrow p$. Assume that its conclusion is false, namely, that n is even. Then $n = 2k$ for some integer k. Then $n^2 = 4k^2 = 2(2k^2)$, so n^2 is even (since it is a multiple of 2). This completes the indirect proof of $q \rightarrow p$.

Since we have shown that both $p \rightarrow q$ and $q \rightarrow p$ are true, we have shown that the theorem is true. ■

Sometimes a theorem states that several propositions are equivalent. Such a theorem states that propositions $p_1, p_2, p_3, \ldots, p_n$ are equivalent. This can be written as

$$p_1 \leftrightarrow p_2 \leftrightarrow \cdots \leftrightarrow p_n,$$

which states that all n propositions have the same truth values. One way to prove these mutually equivalent is to use the tautology

$$[p_1 \leftrightarrow p_2 \leftrightarrow \cdots \leftrightarrow p_n] \leftrightarrow [(p_1 \rightarrow p_2) \wedge (p_2 \rightarrow p_3) \wedge \cdots \wedge (p_n \rightarrow p_1)].$$

This shows that if the implications $p_1 \rightarrow p_2, p_2 \rightarrow p_3, \ldots, p_n \rightarrow p_1$ can be shown to be true, then the propositions $p_1, p_2, \ldots, p_n$ are all equivalent.

EXAMPLE 22 Prove that when n is an integer, the following three statements are equivalent.

p_1: n **mod** $3 = 1$ or n **mod** $3 = 2$

p_2: n is not divisible by 3

p_3: $n^2 \equiv 1 \pmod 3$

Solution: To show that the statements are equivalent, we can prove that the implications $p_1 \to p_2$, $p_2 \to p_3$, and $p_3 \to p_1$ are true.

We will use a direct proof to show that $p_1 \to p_2$ is true. Assume that n **mod** $3 = 1$ or 2. By the division algorithm, $n = 3q + r$ where $0 \le r < 3$. By the definition of **mod**, we have $r = n$ **mod** 3. Since n is divisible by 3 if and only if $r = 0$, the assumption that n **mod** $3 = 1$ or 2 implies that n is not divisible by 3. This completes the proof that $p_1 \to p_2$ is true.

We have already shown that $p_2 \to p_3$ is true in Example 20.

We will use an indirect proof to show that $p_3 \to p_1$ is true. We assume that the conclusion of this implication is false, namely, that n **mod** 3 is neither 1 nor 2. Since n **mod** 3 equals 0, 1, or 2, we see that n **mod** $3 = 0$. This means that $3 \mid n$, so that $n = 3k$ for some integer k. This implies that $n^2 = 9k^2 = 3(3k^2)$, which shows that $n^2 \equiv 0 \pmod 3$, so that p_3 is false. This completes the indirect proof that $p_3 \to p_1$, and it also completes the proof of the theorem. ■

THEOREMS AND QUANTIFIERS

Many theorems are stated as propositions that involve quantifiers. A variety of methods are used to prove theorems that are quantifications. We will describe some of the most important of these here.

Many theorems are assertions that objects of a particular type exist. A theorem of this type is a proposition of the form $\exists x\, P(x)$, where P is a predicate. A proof of a proposition of the form $\exists x\, P(x)$ is called an **existence proof.** There are several ways to prove a theorem of this type. Sometimes an existence proof of $\exists x\, P(x)$ can be given by finding an element a such that $P(a)$ is true. Such an existence proof is called **constructive.** It is also possible to give an existence proof that is **nonconstructive;** that is, we do not find an element a such that $P(a)$ is true, but rather prove that $\exists x\, P(x)$ is true in some other way. One common method of giving a nonconstructive existence proof is to use proof by contradiction and show that the negation of the existential quantification implies a contradiction. The concept of a constructive existence proof is illustrated by the following example.

EXAMPLE 23 A Constructive Existence Proof Show that there are n consecutive composite positive integers for every positive integer n. Note that this asks for proof of the quantification: $\forall n\, \exists x (x + i \text{ is composite for } i = 1, 2, \ldots, n)$.

Solution: Let

$$x = (n + 1)! + 1.$$

Consider the integers

$$x + 1, x + 2, \ldots, x + n.$$

Note that $i+1$ divides $x+i = (n+1)!+(i+1)$ for $i = 1, 2, \ldots, n$. Hence, n consecutive composite positive integers have been given. Note that in the solution a number x such that $x+i$ is composite for $i = 1, 2, \ldots, n$ has been produced. Hence, this is an example of a constructive existence proof. ■

Remark: The proof in Example 23 can be found in the works of the ancient Greek mathematician Euclid.

An example of a nonconstructive existence proof is given next.

EXAMPLE 24 A Nonconstructive Existence Proof Show that for every positive integer n there is a prime greater than n. This problem asks for a proof of an existential quantification, namely, $\exists x\, Q(x)$, where $Q(x)$ is the proposition "x is prime and x is greater than n," and the universe of discourse is the set of positive integers.

Solution: Let n be a positive integer. To show that there is a prime greater than n, consider the integer $n!+1$. Since every integer has a prime factor, there is at least one prime dividing $n!+1$. (One possibility is that $n!+1$ is already prime.) Note that when $n!+1$ is divided by an integer less than or equal to n, the remainder equals 1. Hence, any prime factor of this integer must be greater than n. This proves the result. This argument is a nonconstructive existence proof because a prime larger than n has not been produced. It has simply been shown that one must exist. ■

Suppose a statement of the form $\forall x\, P(x)$ is false. How can we show this? Recall that the propositions $\neg\forall x\, P(x)$ and $\exists x\, \neg P(x)$ are equivalent. This means that if we find an element a such that $P(a)$ is false, then we have shown that $\exists x\, \neg P(x)$ is true, which means that $\forall x\, P(x)$ is false. An element a for which $P(a)$ is false is called a **counterexample**. Note that only one counterexample needs to be found to show that $\forall x\, P(x)$ is false.

EXAMPLE 25 Show that the assertion "All primes are odd" is false.

Solution: The statement "All primes are odd" is a universal quantification, namely,

$$\forall x\, O(x),$$

where $O(x)$ is the proposition "x is odd," and the universe of discourse is the set of primes. Note that $x = 2$ is a counterexample, since 2 is a prime number that is even. Hence, the statement "All prime numbers are odd" is false. ■

It is a common mistake to assume that one or more examples establish the truth of a statement. No matter how many examples there are where $P(x)$ is true, the universal quantification $\forall x P(x)$ may still be false. Consider the following example.

EXAMPLE 26 Is $n^2 - n + 41$ prime for all nonnegative integers n? That is, is the statement $\forall n\, P(n)$ a theorem, where $P(n)$ is the statement "$n^2 - n + 41$ is prime" and the universe of discourse is the set of nonnegative integers?

Solution: To determine whether $n^2 - n + 41$ is prime for all nonnegative integers, we might begin by examining whether it is prime for the smallest nonnegative integers. We find that $n^2 - n + 41$ is prime for all nonnegative integers not exceeding 40 (as the reader can verify). However, if we decided this was enough checking, we would come to the wrong conclusion. It is not true that $n^2 - n + 41$ is prime for all nonnegative integers. When $n = 41$, it is composite (as the reader should verify). ■

Example 26 helps illustrate the crucial point that a statement may not be true, even though there are many examples for which it is true.

THE HALTING PROBLEM

web

We will now describe a proof of one of the most famous theorems in computer science. We will show that there is a problem which cannot be solved using any procedure. That is, we will show there are unsolvable problems, as was mentioned in Section 2.2. The problem we will study is the **halting problem**. It asks whether there is a procedure that does the following: It takes as input a computer program and input to the program and determines whether the program will eventually stop when run with this input. It would be convenient to have such a procedure, if it existed. Certainly being able to test whether a program entered into an infinite loop would be helpful when writing and debugging programs. However, in 1936 Alan Turing showed that no such procedure exists (see his biography in Section 10.4).

Before we present a proof that the halting problem is unsolvable, first note that we cannot simply run a program and observe what it does to determine whether it terminates when run with the given input. If the program halts, we have our answer, but if it is still running after any fixed length of time has elapsed, we do not know whether it will never halt or we just did not wait long enough for it to terminate. After all, it is not hard to design a program that will stop only after more than a billion years has elapsed.

We will describe Turing's proof that the halting problem is unsolvable; it is a proof by contradiction. (The reader should note that our proof is not completely rigorous, since we have not explicitly defined what a procedure is. To remedy this, the concept of a Turing machine is needed. This concept is introduced in Section 10.5.)

Proof: Assume there is a solution to the halting problem, a procedure called $H(P, I)$. The procedure $H(P, I)$ takes two inputs, one a program P and the other I, an input to the program P. $H(P, I)$ generates the string "halt" as output if H determines that P stops when given I as input. Otherwise, $H(P, I)$ generates the string "loops forever" as output. We will now derive a contradiction.

When a procedure is coded, it is expressed as a string of characters; this string can be interpreted as a sequence of bits. This means that a program itself can be used as data. Therefore a program can be thought of as input to another program, or even itself. Hence, H can take a program P as both of its inputs, which are a program and input to this program. H should be able to determine if P will halt when it is given a copy of itself as input.

To show that no procedure H exists which solves the halting problem, we construct a simple procedure $K(P)$, which works as follows, making use of the output $H(P, P)$. If the output of $H(P, P)$ is "loops forever," which means that P loops forever when given a copy of itself as input, then $K(P)$ halts. If the output of $H(P, P)$ is "halt," which means that P halts when given a copy of itself as input, then $K(P)$ loops forever. That is, $K(P)$ does the opposite of what the output of $H(P, P)$ specifies. (See Figure 1.)

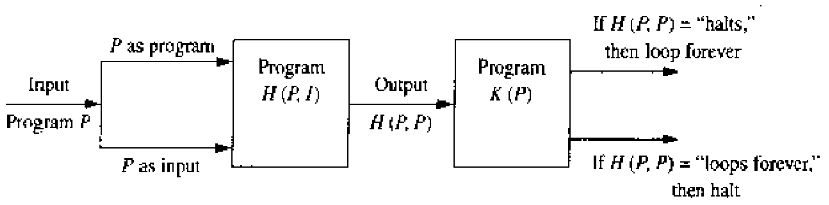

P as program
Input
Program P
P as input
Program
H (P, I)
Output
H (P, P)
Program
K (P)
If H (P, P) = "halts," then loop forever
If H (P, P) = "loops forever," then halt

FIGURE 1

Now suppose we provide K as input to K. We note that if the output of $H(K, K)$ is "loops forever," then by the definition of K we see that $K(K)$ halts. Otherwise, if the output of $H(K, K)$ is "halt," then by the definition of K we see that $K(K)$ loops forever, in violation of what H tells us. In both cases, we have a contradiction.

Thus, H cannot always give the correct answers. Consequently, there is no procedure that solves the halting problem. ☐

SOME COMMENTS ON PROOFS

We described a variety of methods for proving theorems. Observe that no algorithm for proving theorems has been given here. Such a procedure does not exist.

There are many theorems whose proofs are easy to find by directly working through the hypotheses and definitions of the terms in the theorem. However, it is often difficult to prove a theorem without resorting to a clever use of an indirect proof, a proof by contradiction, or some other proof technique. Constructing proofs is an art that can be learned only by trying various lines of attack.

Moreover, many statements that appear to be theorems have resisted the persistent efforts of mathematicians for hundreds of years. For instance, as simple a statement as "every even positive integer greater than 4 is the sum of two primes" has not yet been shown to be true, and no counterexample has been found, although it has been verified for all even positive integers up to 10^{14}. This statement is known as **Goldbach's conjecture** and is one of many assertions in mathematics with a truth value that is unknown.

Exercises

1. What rule of inference is used in each of the following arguments?
   a) Alice is a mathematics major. Therefore, Alice is either a mathematics major or a computer science major.
   b) Jerry is a mathematics major and a computer science major. Therefore, Jerry is a mathematics major.
   c) If it is rainy, then the pool will be closed. It is rainy. Therefore, the pool is closed.

Christian Goldbach (1690–1764). Christian Goldbach was born in Königsberg, Prussia, the city noted for its famous bridge problem (which will be studied in Section 7.5). He became professor of mathematics at the Academy in St. Petersburg in 1725. In 1728 Goldbach went to Moscow to tutor the son of the tsar. He entered the world of politics when, in 1742, he became a staff member in the Russian Ministry of Foreign Affairs. Goldbach is best known for his correspondence with eminent mathematicians, including Euler and Bernoulli, for his famous conjectures in number theory, and for several contributions to analysis.

   d) If it snows today, the university will close. The university is not closed today. Therefore, it did not snow today.

   e) If I go swimming, then I will stay in the sun too long. If I stay in the sun too long, then I will sunburn. Therefore, if I go swimming, then I will sunburn.

2. What rule of inference is used in each of the following arguments?

   a) Kangaroos live in Australia and are marsupials. Therefore, kangaroos are marsupials.

   b) It is either hotter than 100 degrees today or the pollution is dangerous. It is less than 100 degrees outside today. Therefore, the pollution is dangerous.

   c) Linda is an excellent swimmer. If Linda is an excellent swimmer, then she can work as a lifeguard. Therefore, Linda can work as a lifeguard.

   d) Steve will work at a computer company this summer. Therefore, this summer Steve will work at a computer company or he will be a beach bum.

   e) If I work all night on this homework, then I can answer all the exercises. If I answer all the exercises, I will understand the material. Therefore, if I work all night on this homework, then I will understand the material.

3. Construct an argument using rules of inference to show that the hypotheses "Randy works hard," "If Randy works hard, then he is a dull boy," and "If Randy is a dull boy, then he will not get the job" imply the conclusion "Randy will not get the job."

4. Construct an argument using rules of inference to show that the hypotheses "If it does not rain or if it is not foggy, then the sailing race will be held and the lifesaving demonstration will go on," "If the sailing race is held, then the trophy will be awarded," and "The trophy was not awarded" imply the conclusion "It rained."

5. What rules of inference are used in the following famous argument? "All men are mortal. Socrates is a man. Therefore, Socrates is mortal."

6. What rules of inference are used in the following argument? "No man is an island. Manhattan is an island. Therefore, Manhattan is not a man."

7. For each of the following sets of premises, what relevant conclusion or conclusions can be drawn? Explain the rules of inference used to obtain each conclusion from the premises.

   a) "If I take the day off, it either rains or snows." "I took Tuesday off or I took Thursday off." "It was sunny on Tuesday." "It did not snow on Thursday."

   b) "If I eat spicy foods, then I have strange dreams." "I have strange dreams if there is thunder while I sleep." "I did not have strange dreams."

   c) "I am either clever or lucky." "I am not lucky." "If I am lucky, then I will win the lottery."

   d) "Every computer science major has a personal computer." "Ralph does not have a personal computer." "Ann has a personal computer."

   e) "What is good for corporations is good for the United States." "What is good for the United States is good for you." "What is good for corporations is for you to buy lots of stuff."

   f) "All rodents gnaw their food." "Mice are rodents." "Rabbits do not gnaw their food." "Bats are not rodents."

8. For each of the following sets of premises, what relevant conclusion or conclusions can be drawn? Explain the rules of inference used to obtain each conclusion from the premises.

   a) "If I play hockey, then I am sore the next day." "I use the whirlpool if I am sore." "I did not use the whirlpool."

   b) "If I work, it is either sunny or partly sunny." "I worked last Monday or I worked last Friday." "It was not sunny on Tuesday." "It was not partly sunny on Friday."

   c) "All insects have six legs." "Dragonflies are insects." "Spiders do not have six legs." "Spiders eat dragonflies."

   d) "Every student has an Internet account." "Homer does not have an Internet account." "Maggie has an Internet account."

   e) "All foods that are healthy to eat do not taste good." "Tofu is healthy to eat." "You only eat what tastes good." "You do not eat tofu." "Cheeseburgers are not healthy to eat."

   f) "I am either dreaming or hallucinating." "I am not dreaming." "If I am hallucinating, I see elephants running down the road."

9. For each of the following arguments, explain which rules of inference are used for each step.

   a) "Doug, a student in this class, knows how to write programs in JAVA. Everyone who knows how to write programs in JAVA can get a high-paying job. Therefore, someone in this class can get a high-paying job."

   b) "Somebody in this class enjoys whale watching. Every person who enjoys whale watching cares about ocean pollution. Therefore, there is a person in this class who cares about ocean pollution."

   c) "Each of the 93 students in this class owns a personal computer. Everyone who owns a personal computer can use a word processing program. Therefore, Zeke, a student in this class, can use a word processing program."

   d) "Everyone in New Jersey lives within 50 miles of the ocean. Someone in New Jersey has never seen the ocean. Therefore, someone who lives within 50 miles of the ocean has never seen the ocean."

10. For each of the following arguments, explain which rules of inference are used for each step.
  - **a)** "Linda, a student in this class, owns a red convertible. Everyone who owns a red convertible has gotten at least one speeding ticket. Therefore, someone in this class has gotten a speeding ticket."
  - **b)** "Each of five roommates, Melissa, Aaron, Ralph, Veneesha, and Keeshawn, has taken a course in discrete mathematics. Every student who has taken a course in discrete mathematics can take a course in algorithms. Therefore, all five roommates can take a course in algorithms next year."
  - **c)** "All movies produced by John Sayles are wonderful. John Sayles produced a movie about coal miners. Therefore, there is a wonderful movie about coal miners."
  - **d)** "There is someone in this class who has been to France. Everyone who goes to France visits the Louvre. Therefore, someone in this class has visited the Louvre."

11. Determine whether each of the following arguments is valid. If an argument is correct, what rule of inference is being used? If it is not, what fallacy occurs?
  - **a)** If n is a real number such that $n > 1$, then $n^2 > 1$. Suppose that $n^2 > 1$. Then $n > 1$.
  - **b)** The number $\log_2 3$ is irrational if it is not the ratio of two integers. Therefore, since $\log_2 3$ cannot be written in the form a/b where a and b are integers, it is irrational.
  - **c)** If n is a real number with $n > 3$, then $n^2 > 9$. Suppose that $n^2 \leq 9$. Then $n \leq 3$.
  - **d)** A positive integer is either a perfect square or it has an even number of positive integer divisors. Suppose that n is a positive integer that has an odd number of positive integer divisors. Then n is a perfect square.
  - **e)** If n is a real number with $n > 2$, then $n^2 > 4$. Suppose that $n \leq 2$. Then $n^2 \leq 4$.

12. The following argument is an incorrect proof of the theorem "If n^2 is not divisible by 3, then n is not divisible by 3." The reason it is incorrect is that circular reasoning has been used. Where has the error in reasoning been made?

> If n^2 is not divisible by 3, then n^2 does not equal $3k$ for some integer k. Hence, n does not equal $3l$ for some integer l. Therefore, n is not divisible by 3.

13. Prove the proposition $P(0)$, where $P(n)$ is the proposition "If n is a positive integer greater than 1, then $n^2 > n$." What kind of proof did you use?

14. Prove the proposition $P(1)$, where $P(n)$ is the proposition "If n is a positive integer, then $n^2 \geq n$." What kind of proof did you use?

15. Let $P(n)$ be the proposition "If a and b are positive real numbers, then $(a+b)^n \geq a^n + b^n$." Prove that $P(1)$ is true. What kind of proof did you use?

16. Prove that the square of an even number is an even number using
  - **a)** a direct proof.
  - **b)** an indirect proof.
  - **c)** a proof by contradiction.

17. Prove that if n is an integer and $n^3 + 5$ is odd, then n is even using
  - **a)** an indirect proof.
  - **b)** a proof by contradiction.

18. Prove that if n is an integer and $3n + 2$ is even, then n is even using
  - **a)** an indirect proof.
  - **b)** a proof by contradiction.

19. Prove that the sum of two odd integers is even.

20. Prove that the sum of two rational numbers is rational.

21. Prove that the sum of an irrational number and a rational number is irrational using a proof by contradiction.

22. Prove that the product of two rational numbers is rational.

23. Prove or disprove that the product of two irrational numbers is irrational.

24. Prove or disprove that the product of a nonzero rational number and an irrational number is irrational.

***25.** Prove or disprove that $n^2 - 79n + 1601$ is prime whenever n is a positive integer.

26. Prove or disprove that $2^n + 1$ is prime for all nonnegative integers n.

27. Show that $\sqrt[3]{3}$ is irrational.

***28.** Show that $\sqrt{n}$ is irrational if n is a positive integer that is not a perfect square.

29. Prove that if x and y are real numbers, then $\max(x, y) + \min(x, y) = x + y$. (*Hint:* Use a proof by cases, with the two cases corresponding to $x \geq y$ and $x < y$, respectively.)

30. Prove that the square of an integer not divisible by 5 leaves a remainder of 1 or 4 when divided by 5. (*Hint:* Use a proof by cases, where the cases correspond to the possible remainders for the integer when it is divided by 5.)

31. Prove that if x and y are real numbers, then $|x| + |y| \geq |x + y|$ (where $|x|$ represents the absolute value of x, which equals x if $x \geq 0$ and equals $-x$ if $x \leq 0$).

32. Use a proof by cases to show that $\lfloor n/2 \rfloor \lceil n/2 \rceil = \lfloor n^2/4 \rfloor$ for all integers n.

33. Use a proof by cases to show that $\min(a, \min(b, c)) = \min(\min(a, b), c)$ whenever a, b, and c are real numbers.

34. Prove that if n is a positive integer, then n is even if and only if $7n + 4$ is even.

35. Prove that if n is a positive integer, then n is odd if and only if $5n + 6$ is odd.

36. Prove that $m^2 = n^2$ if and only if $m = n$ or $m = -n$.

***37.** Let p be prime. Prove that $a^2 \equiv b^2 \pmod{p}$ if and only if $a \equiv b \pmod{p}$ or $a \equiv -b \pmod{p}$.

38. Prove or disprove that $n^2 - 1$ is composite whenever n is a positive integer greater than 1.

39. Prove or disprove that if m and n are integers such that $mn = 1$, then either $m = 1$ and $n = 1$, or else $m = -1$ and $n = -1$.

40. Prove or disprove that $a \textbf{ mod } m + b \textbf{ mod } m = (a + b) \textbf{ mod } m$ whenever m is a positive integer.

41. Prove or disprove that every positive integer can be written as the sum of the squares of two integers.

42. Prove that if n is a positive integer such that the sum of its divisors is $n + 1$, then n is prime. What kind of proof did you use?

43. Prove or disprove each of the following statements about the floor and ceiling functions.

a) $\lceil \lfloor x \rfloor \rceil = \lfloor x \rfloor$ for all real numbers x.

b) $\lfloor 2x \rfloor = 2\lfloor x \rfloor$ whenever x is a real number.

c) $\lceil x \rceil + \lceil y \rceil - \lceil x + y \rceil = 0$ or 1 whenever x and y are real numbers.

d) $\lceil xy \rceil = \lceil x \rceil \lceil y \rceil$ for all real numbers x and y.

e) $\left\lceil \frac{x}{2} \right\rceil = \left\lfloor \frac{x+1}{2} \right\rfloor$ for all real numbers x.

44. Prove or disprove each of the following statements about the floor and ceiling functions.

a) $\lfloor \lceil x \rceil \rfloor = \lceil x \rceil$ for all real numbers x.

b) $\lfloor x + y \rfloor = \lfloor x \rfloor + \lfloor y \rfloor$ for all real numbers x and y.

c) $\lceil \lceil x/2 \rceil /2 \rceil = \lceil x/4 \rceil$ for all real numbers x.

d) $\lfloor \sqrt{\lceil x \rceil} \rfloor = \lfloor \sqrt{x} \rfloor$ for all real numbers x.

e) $\lfloor x \rfloor + \lfloor y \rfloor + \lfloor x + y \rfloor \le \lfloor 2x \rfloor + \lfloor 2y \rfloor$ for all real numbers x and y.

45. Prove that if x is a positive real number, then

a) $\lfloor \sqrt{\lfloor x \rfloor} \rfloor = \lfloor \sqrt{x} \rfloor$.

b) $\lceil \sqrt{\lceil x \rceil} \rceil = \lceil \sqrt{x} \rceil$.

46. Prove that if m and n are positive integers and x is a real number, then

$$\left\lfloor \frac{\lfloor x \rfloor + n}{m} \right\rfloor = \left\lfloor \frac{x + n}{m} \right\rfloor .$$

***47.** Prove that if m is a positive integer and x is a real number, then

$$\lfloor mx \rfloor = \lfloor x \rfloor + \left\lfloor x + \frac{1}{m} \right\rfloor + \left\lfloor x + \frac{2}{m} \right\rfloor + \cdots + \left\lfloor x + \frac{m-1}{m} \right\rfloor .$$

****48.** Show that if a and b are positive irrational numbers such that $1/a + 1/b = 1$, then every positive integer can be uniquely expressed as either $\lfloor ka \rfloor$ or $\lfloor kb \rfloor$ for some positive integer k.

49. Prove that at least one of the real numbers $a_1, a_2, \ldots, a_n$ is greater than or equal to the average of these numbers. What kind of proof did you use?

***50.** Use Exercise 49 to show that if the first 10 positive integers are placed around a circle, in any order, there exist three integers in consecutive locations around the circle that have a sum greater than or equal to 17.

51. Prove that if n is an integer, the following four statements are equivalent: (i) n is even, (ii) $n + 1$ is odd, (iii) $3n + 1$ is odd, (iv) $3n$ is even.

52. Prove that if n is an integer, the following three statements are equivalent: (i) 5 divides n, (ii) 5 divides n^2, (iii) $n^2 \not\equiv \pm 1 \pmod 5$.

53. Prove or disprove that there are three consecutive odd positive integers that are primes, that is, odd primes of the form p, $p + 2$, and $p + 4$.

54. Prove or disprove that given a positive integer n, there are n consecutive odd positive integers that are primes.

55. Which rules of inference are used to establish the conclusion of Lewis Carroll's argument described in Example 20 of Section 1.3?

56. Which rules of inference are used to establish the conclusion of Lewis Carroll's argument described in Example 21 of Section 1.3?

57. Give a constructive proof of the proposition: "For every positive integer n there is an integer divisible by more than n primes."

58. Find a counterexample to the proposition: "For every prime number n, $n + 2$ is prime."

***59.** Prove that there are infinitely many primes congruent to 3 modulo 4. Is your proof constructive or nonconstructive? (*Hint:* One approach is to assume that there are only finitely many such primes $p_1, p_2, \ldots, p_n$. Let $q = 4p_1p_2 \cdots p_n + 3$. Show that q must have a prime factor congruent to 3 modulo 4 not among the n primes $p_1, p_2, \ldots, p_n$.)

60. Prove or disprove that if $p_1, p_2, \ldots, p_n$ are the n smallest primes, then $p_1p_2 \cdots p_n + 1$ is prime.

61. Show that the propositions p_1, p_2, p_3, p_4, and p_5 can be shown to be equivalent by proving that the implications $p_1 \to p_4$, $p_3 \to p_1$, $p_4 \to p_2$, $p_2 \to p_5$, and $p_5 \to p_3$ are true.

62. Prove or disprove that if a and b are rational numbers, then a^b is also rational.

63. Prove that there are irrational numbers a and b such that a^b is rational. Is your proof constructive or nonconstructive? [*Hint:* Let $a = \sqrt{2}$ and $b = \sqrt{2}$. Show that either a^b or $(a^b)^b$ is rational.]

64. Prove that an 8×8 chessboard can be completely covered using dominos (1×2 pieces).

***65.** Prove that it is impossible to cover completely with dominos the 8×8 chessboard with two squares at opposite corners of the board removed.

***66.** The Logic Problem, taken from WFF'N PROOF, The Game of Logic, has the following two assumptions:

1. "Logic is difficult or not many students like logic."

2. "If mathematics is easy, then logic is not difficult."

By translating these assumptions into statements involving propositional variables and logical connectives, determine whether each of the following are valid conclusions of these assumptions:

a) That mathematics is not easy, if many students like logic.

b) That not many students like logic, if mathematics is not easy.
c) That mathematics is not easy or logic is difficult.
d) That logic is not difficult or mathematics is not easy.
e) That if not many students like logic, then either mathematics is not easy or logic is not difficult.

***67.** Determine whether the following argument, taken from Backhouse [Ba86], is valid.

> If Superman were able and willing to prevent evil, he would do so. If Superman were unable to prevent evil, he would be impotent; if he were unwilling to prevent evil, he would be malevolent. Superman does not prevent evil. If Superman exists, he is neither impotent nor malevolent. Therefore, Superman does not exist.

Resolution is a proof method used extensively in artificial intelligence and by automated proof programs. Resolution is based on the rule of inference derived from the tautology $((p \lor q) \land (\neg p \lor r)) \to (q \lor r)$.

68. Use the resolution rule of inference to prove the statement "You will win the lottery or you will be promoted," given the hypotheses "You will quit your job or you will win the lottery," "You will not quit your job or you will find a better job," and "You will not find a better job or you will be promoted."

3.2

Mathematical Induction

INTRODUCTION

web

What is a formula for the sum of the first n positive odd integers? The sums of the first n positive odd integers for $n = 1, 2, 3, 4, 5$ are

$$1 = 1, \qquad 1 + 3 = 4, \qquad 1 + 3 + 5 = 9,$$
$$1 + 3 + 5 + 7 = 16, \qquad 1 + 3 + 5 + 7 + 9 = 25.$$

From these values it is reasonable to guess that the sum of the first n positive odd integers is n^2. We need a method to *prove* that this *guess* is correct, if in fact it is.

Mathematical induction is an extremely important proof technique that can be used to prove assertions of this type. As we will see in this section and in subsequent chapters, mathematical induction is used extensively to prove results about a large variety of discrete objects. For example, it is used to prove results about the complexity of algorithms, the correctness of certain types of computer programs, theorems about graphs and trees, as well as a wide range of identities and inequalities.

In this section we will describe how mathematical induction can be used and why it is a valid proof technique. It is extremely important to note that mathematical induction can be used only to prove results obtained in some other way. It is *not* a tool for discovering formulae or theorems.

THE WELL-ORDERING PROPERTY

The validity of mathematical induction follows from the following fundamental axiom about the set of integers.

The Well-Ordering Property Every nonempty set of nonnegative integers has a least element.

The well-ordering property can often be used directly in proofs.

EXAMPLE 1 Use the well-ordering property to prove the division algorithm. Recall that the division algorithm states that if a is an integer and d is a positive integer, then there are unique integers q and r with $0 \leq r < d$ and $a = dq + r$.

Solution: Let S be the set of nonnegative integers of the form $a - dq$ where q is an integer. This set is nonempty since $-dq$ can be made as large as desired (taking q to be a negative integer with large absolute value). By the well-ordering property S has a least element $r = a - dq_0$.

The integer r is nonnegative. It is also the case that $r < d$. If it were not, then there would be a smaller nonnegative element in S, namely, $a - d(q_0 + 1)$. To see this, suppose that $r \geq d$. Since $a = dq_0 + r$, it follows that $a - d(q_0 + 1) = (a - dq_0) - d = r - d \geq 0$. Consequently, there are integers q and r with $0 \leq r < d$. The proof that q and r are unique is left as an exercise for the reader. ■

MATHEMATICAL INDUCTION

Many theorems state that $P(n)$ is true for all positive integers n, where $P(n)$ is a propositional function, such as the statement that $1 + 2 + \cdots + n = n(n+1)/2$ or the statement that $n \leq 2^n$. Mathematical induction is a technique for proving theorems of this kind. In other words, mathematical induction is used to prove propositions of the form $\forall n\, P(n)$, where the universe of discourse is the set of positive integers.

A proof by mathematical induction that $P(n)$ is true for every positive integer n consists of two steps:

1. *Basis step.* The proposition $P(1)$ is shown to be true.
2. *Inductive step.* The implication $P(n) \rightarrow P(n + 1)$ is shown to be true for every positive integer n.

Here, the statement $P(n)$ for a fixed positive integer n is called the **inductive hypothesis.** When we complete both steps of a proof by mathematical induction, we have proved that $P(n)$ is true for all positive integers n; that is, we have shown that $\forall n\, P(n)$ is true.

Expressed as a rule of inference, this proof technique can be stated as

$$[P(1) \wedge \forall n (P(n) \rightarrow P(n + 1))] \rightarrow \forall n\, P(n).$$

Since mathematical induction is such an important technique, it is worthwhile to explain in detail the steps of a proof using this technique. The first thing we do to prove that $P(n)$ is true for all positive integers n is to show that $P(1)$ is true. This amounts to showing that the particular statement obtained when n is replaced by 1 in $P(n)$ is true. Then we must show that $P(n) \rightarrow P(n + 1)$ is true for every positive integer n. To prove that this implication is true for every positive integer n, we need to show that $P(n + 1)$ cannot be false when $P(n)$ is true. This can be accomplished by assuming that $P(n)$ is true and showing that *under this hypothesis* $P(n + 1)$ must also be true.

Remark: In a proof by mathematical induction it is *not* assumed that $P(n)$ is true for all positive integers! It is only shown that *if it is assumed* that $P(n)$ is true, then $P(n + 1)$ is also true. Thus, a proof by mathematical induction is not a case of begging the question, or circular reasoning.

When we use mathematical induction to prove a theorem, we first show that $P(1)$ is true. Then we know that $P(2)$ is true, since $P(1)$ implies $P(2)$. Further, we know that $P(3)$ is true, since $P(2)$ implies $P(3)$. Continuing along these lines, we see that $P(k)$ is true, for any positive integer k.

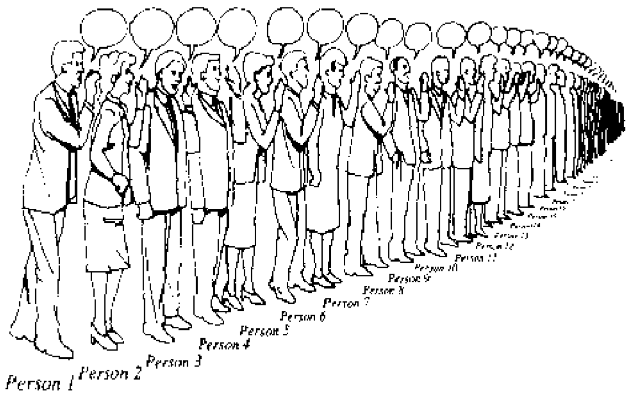


FIGURE 1 **People Telling Secrets.**

There are several useful illustrations of mathematical induction that can help you remember how this principle works. One of these involves a line of people, person one, person two, and so on. A secret is told to person one, and each person tells the secret to the next person in line, if the former person hears it. Let $P(n)$ be the proposition that person n knows the secret. Then $P(1)$ is true, since the secret is told to person one; $P(2)$ is true, since person one tells person two the secret; $P(3)$ is true, since person two tells person three the secret; and so on. By the principle of mathematical induction, every person in line learns the secret. This is illustrated in Figure 1. (Of course, it has been assumed that each person relays the secret in an unchanged manner to the next person, which is usually not true in real life.)

Another way to illustrate the principle of mathematical induction is to consider an infinite row of dominos, labeled $1, 2, 3, \ldots, n$, where each domino is standing up. Let $P(n)$ be the proposition that domino n is knocked over. If the first domino is knocked over—i.e., if $P(1)$ is true—and if, whenever the nth domino is knocked over, it also knocks the $(n+1)$th domino over—i.e., if $P(n) \rightarrow P(n+1)$ is true—then all the dominos are knocked over. This is illustrated in Figure 2.

Why Mathematical Induction Is Valid Why is mathematical induction a valid proof technique? The reason comes from the well-ordering property. Suppose we know that $P(1)$ is true and that the proposition $P(n) \rightarrow P(n+1)$ is true for all positive integers n. To show that $P(n)$ must be true for all positive integers, assume that there is at least one positive integer for which $P(n)$ is false. Then the set S of positive integers for which $P(n)$ is false is nonempty. Thus, by the well-ordering property, S has a least element, which will be denoted by k. We know that k cannot be 1, since $P(1)$ is true. Since k is positive and greater than 1, $k-1$ is a positive integer. Furthermore, since $k-1$ is less than k, it is not in S, so $P(k-1)$ must be true. Since the implication $P(k-1) \rightarrow P(k)$ is also true, it must be the case that $P(k)$ is true. This contradicts the choice of k. Hence, $P(n)$ must be true for every positive integer n.

Historical note: The first known use of mathematical induction is in the work of the sixteenth-century mathematician Francesco Maurolico (1494–1575). Maurolico wrote extensively on the works of classical mathematics and made many contributions to geometry and optics. In his book *Arithmeticorum Libri Duo*, Maurolico presented a variety of properties of the integers together with proofs of these properties. To prove some of these properties he devised the method of mathematical induction. His first use of mathematical induction in this book was to prove that the sum of the first n odd positive integers equals n^2.

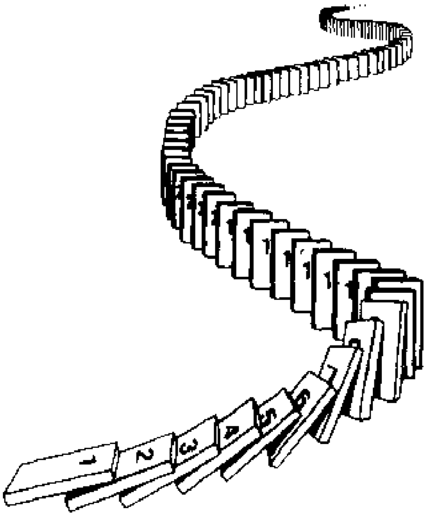


FIGURE 2 **Illustrating How Mathematical Induction Works Using Dominos.**

EXAMPLES OF PROOFS BY MATHEMATICAL INDUCTION

Web

We will use a variety of examples to illustrate how theorems are proved using mathematical induction. We begin by proving a formula for the sum of the first n odd positive integers. (Many theorems proved in this section via mathematical induction can be proved using different methods. However, it is worthwhile to try to prove a theorem in more than one way, since one method of attack may succeed whereas another approach may not.)

EXAMPLE 2 Use mathematical induction to prove that the sum of the first n odd positive integers is n^2.

Solution: Let $P(n)$ denote the proposition that the sum of the first n odd positive integers is n^2. We must first complete the basis step; that is, we must show that $P(1)$ is true. Then we must carry out the inductive step; that is, we must show that $P(n+1)$ is true when $P(n)$ is assumed to be true.

BASIS STEP: $P(1)$ states that the sum of the first one odd positive integers is 1^2. This is true since the sum of the first odd positive integer is 1.

INDUCTIVE STEP: To complete the inductive step we must show that the proposition $P(n) \rightarrow P(n+1)$ is true for every positive integer n. To do this, suppose that $P(n)$ is true for a positive integer n; that is,

$$1 + 3 + 5 + \cdots + (2n - 1) = n^2.$$

[Note that the nth odd positive integer is $(2n - 1)$, since this integer is obtained by adding 2 a total of $n - 1$ times to 1.] We must show that $P(n+1)$ is true, assuming that $P(n)$ is true. Note that $P(n+1)$ is the statement that

$$1 + 3 + 5 + \cdots + (2n - 1) + (2n + 1) = (n + 1)^2.$$

So, assuming that $P(n)$ is true, it follows that

$$\begin{aligned} 1 + 3 + 5 + \cdots + (2n-1) + (2n+1) &= [1 + 3 + \cdots + (2n-1)] + (2n+1) \\ &= n^2 + (2n+1) \\ &= n^2 + 2n + 1 \\ &= (n+1)^2. \end{aligned}$$

This shows that $P(n+1)$ follows from $P(n)$. Note that we used the inductive hypothesis $P(n)$ in the second equality to replace the sum of the first n odd positive integers by n^2.

Since $P(1)$ is true and the implication $P(n) \rightarrow P(n+1)$ is true for all positive integers n, the principle of mathematical induction shows that $P(n)$ is true for all positive integers n. ■

The next example uses the principle of mathematical induction to prove an inequality.

EXAMPLE 3 Use mathematical induction to prove the inequality

$$n < 2^n$$

for all positive integers n.

Solution: Let $P(n)$ be the proposition "$n < 2^n$."

BASIS STEP: $P(1)$ is true, since $1 < 2^1 = 2$.

INDUCTIVE STEP: Assume that $P(n)$ is true for the positive integer n. That is, assume that $n < 2^n$. We need to show that $P(n+1)$ is true. That is, we need to show that $n + 1 < 2^{n+1}$. Adding 1 to both sides of $n < 2^n$, and then noting that $1 \leq 2^n$, gives

$$n + 1 < 2^n + 1 \leq 2^n + 2^n = 2^{n+1}.$$

We have shown that $P(n+1)$ is true, namely, that $n+1 < 2^{n+1}$, based on the assumption that $P(n)$ is true. The induction step is complete.

Therefore, by the principle of mathematical induction, it has been shown that $n < 2^n$ is true for all positive integers n. ■

We will now use mathematical induction to prove a theorem involving the divisibility of integers.

EXAMPLE 4 Use mathematical induction to prove that $n^3 - n$ is divisible by 3 whenever n is a positive integer.

Solution: To construct the proof, let $P(n)$ denote the proposition: "$n^3 - n$ is divisible by 3."

BASIS STEP: $P(1)$ is true, since $1^3 - 1 = 0$ is divisible by 3.

INDUCTIVE STEP: Assume that $P(n)$ is true; that is, $n^3 - n$ is divisible by 3. We must show that $P(n+1)$ is true. That is, we must show that $(n+1)^3 - (n+1)$ is divisible by 3. Note that

$$\begin{aligned}(n+1)^3 - (n+1) &= (n^3 + 3n^2 + 3n + 1) - (n+1)\\ &= (n^3 - n) + 3(n^2 + n).\end{aligned}$$

Since both terms in this sum are divisible by 3 (the first by the assumption of the inductive step, and the second because it is 3 times an integer), it follows that $(n + 1)^3 - (n + 1)$ is also divisible by 3. This completes the induction step. Thus, by the principle of mathematical induction, $n^3 - n$ is divisible by 3 whenever n is a positive integer. ■

Sometimes we need to show that $P(n)$ is true for $n = k, k+1, k+2, \ldots$, where k is an integer other than 1. We can use mathematical induction to accomplish this as long as we change the basis step. For instance, consider Example 5, which proves that a summation formula is valid for all nonnegative integers, so that we need to prove that $P(n)$ is true for $n = 0, 1, 2, \ldots$.

EXAMPLE 5 Use mathematical induction to show that

$$1 + 2 + 2^2 + \cdots + 2^n = 2^{n+1} - 1$$

for all nonnegative integers n.

Solution: Let $P(n)$ be the proposition that this formula is correct for the integer n.

BASIS STEP: $P(0)$ is true since $2^0 = 1 = 2^1 - 1$.

INDUCTIVE STEP: Assume that $P(n)$ is true. To carry out the inductive step using this assumption, it must be shown that $P(n + 1)$ is true, namely,

$$1 + 2 + 2^2 + \cdots + 2^n + 2^{n+1} = 2^{(n+1)+1} - 1 = 2^{n+2} - 1.$$

Using the inductive hypothesis $P(n)$, it follows that

$$\begin{aligned}1 + 2 + 2^2 + \cdots + 2^n + 2^{n+1} &= (1 + 2 + 2^2 + \cdots + 2^n) + 2^{n+1}\\ &= (2^{n+1} - 1) + 2^{n+1}\\ &= 2 \cdot 2^{n+1} - 1\\ &= 2^{n+2} - 1.\end{aligned}$$

This finishes the inductive step, which completes the proof. ■

As Example 5 demonstrates, to use mathematical induction to show that $P(n)$ is true for $n = k, k+1, k+2, \ldots$, where k is an integer other than 1, we show that $P(k)$ is true (the basis step) and then show that the implication $P(n) \to P(n+1)$ is true for $n = k, k+1, k+2, \ldots$ (the inductive step). Note that k can be negative, zero, or positive. Following the domino analogy we used earlier, imagine that we begin by knocking down the kth domino (the basis step), and as each domino falls, it knocks down the next domino (the inductive step). We leave it to the reader to show that this form of induction is valid (see Exercise 68).

The formula given in Example 5 is a special case of a general result for the sum of the terms of a **geometric progression,** which is a sequence of the form $a, ar, ar^2, \ldots, ar^n, \ldots$, where a and r are real numbers. For instance, the sequence in Example 5 is a geometric progression with $a = 1$ and $r = 2$. Likewise, the sequence $3, 15, 75, \ldots, 3 \cdot 5^n, \ldots$ is a geometric progression with $a = 3$ and $r = 5$. The next example gives a formula for the sum of the first $n + 1$ terms of such a sequence. The proof of this general formula will use mathematical induction.

EXAMPLE 6 Sums of Geometric Progressions Use mathematical induction to prove the following formula for the sum of a finite number of terms of a geometric progression:

$$\sum_{j=0}^{n} ar^j = a + ar + ar^2 + \cdots + ar^n = \frac{ar^{n+1} - a}{r - 1}, \qquad \text{when } r \neq 1.$$

Solution: To prove this formula using mathematical induction, let $P(n)$ be the proposition that the sum of the first $n + 1$ terms of a geometric progression in this formula is correct.

BASIS STEP: $P(0)$ is true, since

$$a = \frac{ar - a}{r - 1}.$$

INDUCTIVE STEP: Assume that $P(n)$ is true. That is, assume

$$a + ar + ar^2 + \cdots + ar^n = \frac{ar^{n+1} - a}{r - 1}.$$

To show that this implies that $P(n + 1)$ is true, add ar^{n+1} to both sides of this equation to obtain

$$a + ar + ar^2 + \cdots + ar^n + ar^{n+1} = \frac{ar^{n+1} - a}{r - 1} + ar^{n+1}.$$

Rewriting the right-hand side of this equation shows that

$$\begin{aligned}\frac{ar^{n+1} - a}{r - 1} + ar^{n+1} &= \frac{ar^{n+1} - a}{r - 1} + \frac{ar^{n+2} - ar^{n+1}}{r - 1} \\ &= \frac{ar^{n+2} - a}{r - 1}.\end{aligned}$$

Combining these equations gives

$$a + ar + ar^2 + \cdots + ar^n + ar^{n+1} = \frac{ar^{n+2} - a}{r - 1}.$$

This shows that if $P(n)$ is true, then $P(n + 1)$ must also be true. This completes the inductive argument and shows that the formula for the sum of the terms of a geometric series is correct. ■

As previously mentioned, the formula in Example 5 is the case of the formula in Example 6 with $a = 1$ and $r = 2$. The reader should verify that putting these values for a and r in the general formula gives the same formula as in Example 5.

An important inequality for the sum of the reciprocals of a set of positive integers will be proved in the next example.

EXAMPLE 7 An Inequality for Harmonic Numbers The *harmonic numbers* H_k, $k = 1, 2, 3, \ldots$, are defined by

$$H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}.$$

For instance,

$$H_4 = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} = \frac{25}{12}.$$

Use mathematical induction to show that

$$H_{2^n} \geq 1 + \frac{n}{2},$$

whenever n is a nonnegative integer.

Solution: To carry out the proof, let $P(n)$ be the proposition that $H_{2^n} \geq 1 + n/2$.

BASIS STEP: $P(0)$ is true, since $H_{2^0} = H_1 = 1 \geq 1 + 0/2$.

INDUCTIVE STEP: Assume that $P(n)$ is true, so that $H_{2^n} \geq 1 + n/2$. It must be shown that $P(n + 1)$, which states that $H_{2^{n+1}} \geq 1 + (n + 1)/2$, must also be true under this assumption. This can be done since

$$\begin{aligned}
H_{2^{n+1}} &= 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{2^n} + \frac{1}{2^n + 1} + \cdots + \frac{1}{2^{n+1}} \\
&= H_{2^n} + \frac{1}{2^n + 1} + \cdots + \frac{1}{2^{n+1}} \\
&\geq \left(1 + \frac{n}{2}\right) + \frac{1}{2^n + 1} + \cdots + \frac{1}{2^{n-1}} && \text{(by the inductive hypothesis)} \\
&\geq \left(1 + \frac{n}{2}\right) + 2^n \cdot \frac{1}{2^{n-1}} && \text{(since there are 2^n terms each not less than $1/2^{n-1}$)} \\
&\geq \left(1 + \frac{n}{2}\right) + \frac{1}{2} \\
&= 1 + \frac{n + 1}{2}.
\end{aligned}$$

This establishes the inductive step of the proof. Thus, the inequality for the harmonic numbers is valid for all nonnegative integers n. ■

Remark: The inequality established here can be used to show that the *harmonic series*

$$1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} + \cdots$$

is a divergent infinite series. This is an important example in the study of infinite series.

The next example shows how mathematical induction can be used to verify a formula for the number of subsets of a finite set.

EXAMPLE 8 The Number of Subsets of a Finite Set Use mathematical induction to show that if S is a finite set with n elements, then S has 2^n subsets. (We will prove this result directly in several ways in Chapter 4.)

Solution: Let $P(n)$ be the proposition that a set with n elements has 2^n subsets.

BASIS STEP: $P(0)$ is true, since a set with zero elements, the empty set, has exactly $2^0 = 1$ subsets, since it has one subset, namely, itself.

INDUCTIVE STEP: Assume that $P(n)$ is true, that is, that every set with n elements has 2^n subsets. It must be shown that under this assumption $P(n + 1)$, which is the statement that every set with $n + 1$ elements has 2^{n+1} subsets, must also be true. To show this, let T be a set with $n + 1$ elements. Then, it is possible to write $T = S \cup \{a\}$ where a is one of the elements of T and $S = T - \{a\}$. The subsets of T can be obtained in the following way. For each subset X of S there are exactly two subsets of T, namely, X and $X \cup \{a\}$. (This is illustrated in Figure 3.) These constitute all the subsets of T and are all distinct. Since there are 2^n subsets of S, there are $2 \cdot 2^n = 2^{n+1}$ subsets of T. This finishes the induction argument. ■

EXAMPLE 9 Show that if n is a positive integer,

$$1 + 2 + \cdots + n = n(n + 1)/2.$$

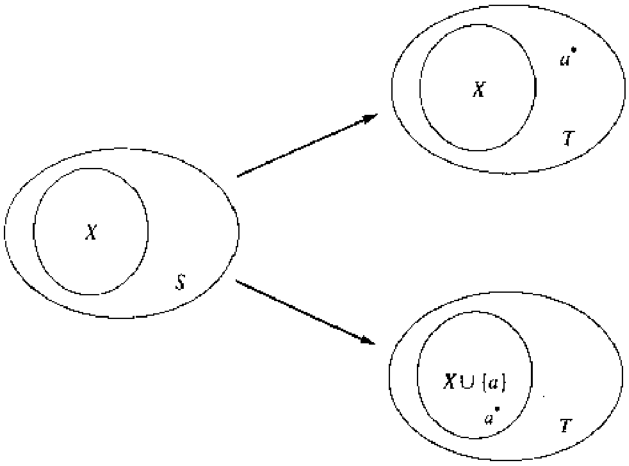


FIGURE 3 **Generating Subsets of a Set with $n + 1$ Elements. Here $T = S \cup \{a\}$.**

Solution: Let $P(n)$ be the proposition that the sum of the first n positive integers is $n(n+1)/2$. We must do two things to prove that $P(n)$ is true for $n = 1, 2, 3, \ldots$. Namely, we must show that $P(1)$ is true and that the implication $P(n)$ implies $P(n+1)$ is true for $n = 1, 2, 3, \ldots$.

BASIS STEP: $P(1)$ is true, since $1 = 1(1+1)/2$.

INDUCTIVE STEP: Assume that $P(n)$ holds so that

$$1 + 2 + \cdots + n = n(n+1)/2.$$

Under this assumption, it must be shown that $P(n+1)$ is true, namely, that

$$1 + 2 + \cdots + n + n + 1 = (n+1)[(n+1)+1]/2 = (n+1)(n+2)/2$$

is also true. Add $n + 1$ to both sides of the equation in $P(n)$ to obtain

$$\begin{aligned} 1 + 2 + \cdots + n + (n+1) &= n(n+1)/2 + (n+1) \\ &= [(n/2) + 1](n+1) \\ &= (n+1)(n+2)/2. \end{aligned}$$

This last equation shows that $P(n+1)$ is true. This completes the inductive step and completes the proof. ■

EXAMPLE 10 Use mathematical induction to prove that $2^n < n!$ for every positive integer n with $n \geq 4$.

Solution: Let $P(n)$ be the proposition that $2^n < n!$.

BASIS STEP: To prove the inequality for $n \geq 4$ requires that the basis step be $P(4)$. Note that $P(4)$ is true, since $2^4 = 16 < 4! = 24$.

INDUCTIVE STEP: Assume that $P(n)$ is true. That is, assume that $2^n < n!$. We must show that $P(n+1)$ is true. That is, we must show that $2^{n+1} < (n+1)!$. Multiplying both sides of the inequality $2^n < n!$ by 2, it follows that

$$\begin{aligned} 2 \cdot 2^n &< 2 \cdot n! \\ &< (n+1) \cdot n! \\ &= (n+1)!. \end{aligned}$$

This shows that $P(n+1)$ is true when $P(n)$ is true. This completes the inductive step of the proof. Hence, it follows that $2^n < n!$ is true for all integers n with $n \geq 4$. ■

EXAMPLE 11 Use mathematical induction to prove the following generalization of one of De Morgan's laws:

$$\overline{\bigcap_{k=1}^{n} A_k} = \bigcup_{k=1}^{n} \overline{A_k},$$

whenever $A_1, A_2, \ldots, A_n$ are subsets of a universal set U and $n \geq 2$.

Solution: Let $P(n)$ be the identity for n sets.

BASIS STEP: The statement $P(2)$ asserts that $\overline{A_1 \cap A_2} = \overline{A_1} \cup \overline{A_2}$. This is one of De Morgan's laws; it was proved in Section 1.5.

INDUCTIVE STEP: Assume that $P(n)$ is true, that is,

$$\overline{\bigcap_{k=1}^{n} A_k} = \bigcup_{k=1}^{n} \overline{A_k}$$

whenever $A_1, A_2, \ldots, A_n$ are subsets of the universal set U. To carry out the inductive step it must be shown that if this equality holds for any n subsets of U, it must also be valid for any $n+1$ subsets of U. Suppose that $A_1, A_2, \ldots, A_n, A_{n+1}$ are subsets of U. When the inductive hypothesis is assumed to hold, it follows that

$$\begin{aligned}
\overline{\bigcap_{k=1}^{n+1} A_k} &= \overline{\left(\bigcap_{k=1}^{n} A_k\right) \cap A_{n+1}} \\
&= \overline{\left(\bigcap_{k=1}^{n} A_k\right)} \cup \overline{A_{n+1}} && \text{(by De Morgan's law)} \\
&= \left(\bigcup_{k=1}^{n} \overline{A_k}\right) \cup \overline{A_{n+1}} && \text{(by the inductive hypothesis)} \\
&= \bigcup_{k=1}^{n+1} \overline{A_k}.
\end{aligned}$$

This completes the proof by induction. ■

The next example illustrates how mathematical induction can be used to prove a result about covering chessboards with pieces shaped like the letter "L."

EXAMPLE 12 Let n be a positive integer. Show that any $2^n \times 2^n$ chessboard with one square removed can be tiled using L-shaped pieces, where these pieces cover three squares at a time, as shown in Figure 4.

Solution: Let $P(n)$ be the proposition that any $2^n \times 2^n$ chessboard with one square removed can be tiled using L-shaped pieces. We can use mathematical induction to prove that $P(n)$ is true for all positive integers n.

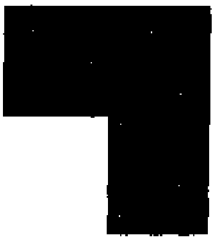


FIGURE 4 An L-Shaped Piece.

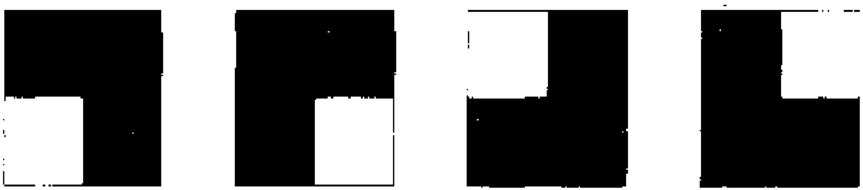


FIGURE 5 **Tiling 2 × 2 Chessboards with One Square Removed.**

BASIS STEP: $P(1)$ is true, since any of the four 2×2 chessboards with one square removed can be tiled using one L-shaped piece, as shown in Figure 5.

INDUCTIVE STEP: Assume that $P(n)$ is true; that is, assume that any $2^n \times 2^n$ chessboard with one square removed can be tiled using L-shaped pieces. It must be shown that under this assumption $P(n+1)$ must also be true; that is, any $2^{n+1} \times 2^{n+1}$ chessboard with one square removed can be tiled using L-shaped pieces.

To see this, consider a $2^{n+1} \times 2^{n+1}$ chessboard with one square removed. Split this chessboard into four chessboards of size $2^n \times 2^n$, by dividing it in half in both directions. This is illustrated in Figure 6. No square has been removed from three of these four chessboards. The fourth $2^n \times 2^n$ chessboard has one square removed, so by the inductive hypothesis, it can be covered by L-shaped pieces. Now temporarily remove the square from each of the other three $2^n \times 2^n$ chessboards that has the center of the original, larger chessboard as one of its corners, as shown in Figure 7. By the inductive hypothesis, each of these three $2^n \times 2^n$ chessboards with a square removed can be tiled by L-shaped pieces. Furthermore, the three squares that were temporarily removed can be covered by one L-shaped piece. Hence, the entire $2^{n+1} \times 2^{n+1}$ chessboard can be tiled with L-shaped pieces. This completes the proof. ■

THE SECOND PRINCIPLE OF MATHEMATICAL INDUCTION

There is another form of mathematical induction that is often useful in proofs. With this form we use the same basis step as before, but we use a different inductive step. We assume that $P(k)$ is true for $k = 1, \ldots, n$ and show that $P(n+1)$ must also be true based on this assumption. This is called the **second principle of mathematical induction.**

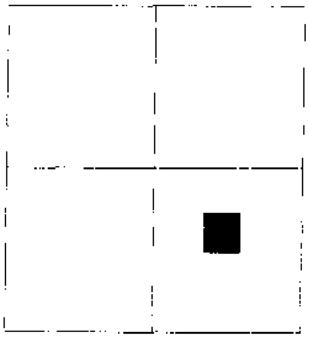


FIGURE 6 **Dividing a $2^{n+1} \times 2^{n+1}$ Chessboard into Four $2^n \times 2^n$ Chessboards.**

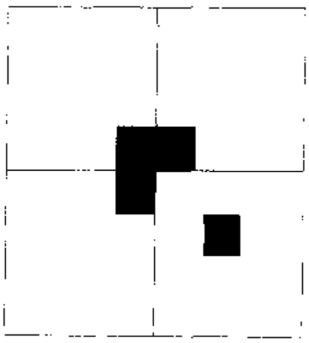


FIGURE 7 **Tiling the $2^{n+1} \times 2^{n+1}$ Chessboard with One Square Removed.**

We summarize the two steps used to show that $P(n)$ is true for all positive integers n:

1. *Basis step.* The proposition $P(1)$ is shown to be true.
2. *Inductive step.* It is shown that $[P(1) \wedge P(2) \wedge \cdots \wedge P(n)] \rightarrow P(n+1)$ is true for every positive integer n.

The two forms of mathematical induction are equivalent; that is, each can be shown to be a valid proof technique assuming the other. We leave it as an exercise for the reader to show this. We now give an example that shows how the second principle of mathematical induction is used.

EXAMPLE 13 Show that if n is an integer greater than 1, then n can be written as the product of primes.

Solution: Let $P(n)$ be the proposition that n can be written as the product of primes.

BASIS STEP: $P(2)$ is true, since 2 can be written as the product of one prime, itself. [Note that $P(2)$ is the first case we need to establish.]

INDUCTIVE STEP: Assume that $P(k)$ is true for all positive integers k with $k \leq n$. To complete the inductive step, it must be shown that $P(n+1)$ is true under this assumption.

There are two cases to consider, namely, when $n+1$ is prime and when $n+1$ is composite. If $n+1$ is prime, we immediately see that $P(n+1)$ is true. Otherwise, $n+1$ is composite and can be written as the product of two positive integers a and b with $2 \leq a \leq b < n+1$. By the induction hypothesis, both a and b can be written as the product of primes. Thus, if $n+1$ is composite, it can be written as the product of primes, namely, those primes in the factorization of a and those in the factorization of b. ∎

Remark: Since 1 is a product of primes, namely, the *empty* product of no primes, we could have started the proof in Example 13 with $P(1)$ as the basis step. We chose not to do this because many people find this confusing.

Note that Example 13 completes the proof of the Fundamental Theorem of Arithmetic, which asserts that every nonnegative integer can be written uniquely as the product of primes in nondecreasing order. We showed in Section 2.5 that an integer has at most one such factorization into primes. Example 13 shows there is at least one such factorization.

Using the principle of mathematical induction, instead of the second principle of mathematical induction, to prove the result in Example 13 is difficult. However, as Example 14 shows, some results can be readily proved using either the principle of mathematical induction or the second principle of mathematical induction.

EXAMPLE 14 Prove that every amount of postage of 12 cents or more can be formed using just 4-cent and 5-cent stamps.

Solution: We will prove this result using the principle of mathematical induction. Then we will present a proof using the second principle of mathematical induction. Let $P(n)$ be the statement that postage of n cents can be formed using 4-cent and 5-cent stamps.

We begin by using the principle of mathematical induction.

BASIS STEP: Postage of 12 cents can be formed using three 4-cent stamps.

INDUCTIVE STEP: Assume that $P(n)$ is true, so that postage of n cents can be formed using 4-cent and 5-cent stamps. If at least one 4-cent stamp was used, replace it with a 5-cent stamp to form postage of $n + 1$ cents. If no 4-cent stamps were used, postage of n cents was formed using just 5-cent stamps. Since $n \geq 12$, at least three 5-cent stamps were used. So, replace three 5-cent stamps with four 4-cent stamps to form postage of $n + 1$ cents. This completes the inductive step, as well as the proof by the principle of mathematical induction.

Next, we will use the second principle of mathematical induction. We will show that postage of 12, 13, 14, and 15 cents can be formed and then show how to get postage of $n + 1$ cents for $n \geq 15$ from postage of $n - 3$ cents.

BASIS STEP: We can form postage of 12, 13, 14, and 15 cents using three 4-cent stamps, two 4-cent stamps and one 5-cent stamp, one 4-cent stamp and two 5-cent stamps, and three 5-cent stamps, respectively.

INDUCTIVE STEP: Let $n \geq 15$. Assume that we can form postage of k cents, where $12 \leq k \leq n$. To form postage of $n + 1$ cents, use the stamps that form postage of $n - 3$ cents together with a 4-cent stamp. This completes the inductive step, as well as the proof by the second principle of mathematical induction.

(There are other ways to approach this problem besides those described here. Can you find a solution that does not use mathematical induction?) ■

Remark: Example 14 shows how we can adapt the second principle of mathematical induction to handle cases where the inductive step is valid only for sufficiently large values of n. In particular, to show that $P(n)$ is true for $n = k, k + 1, k + 2, \ldots$, where k is an integer, we first show that $P(k), P(k + 1), P(k + 2), \ldots, P(l)$ are true (the basis step), and then we show that $[P(k) \wedge P(k + 1) \wedge P(k + 2) \wedge \cdots \wedge P(n)] \rightarrow P(n + 1)$ is true for every integer $n \geq l$ (the inductive step). For example, the basis step of the second proof in the solution of Example 14 shows that $P(12)$, $P(13)$, $P(14)$, and $P(15)$ are true. We need to prove these cases separately since the inductive step, which shows that $[P(12) \wedge P(13) \wedge \cdots \wedge P(n)] \rightarrow P(n + 1)$, only holds when $n \geq 15$.

We will discuss two important applications of mathematical induction in the following sections. The first involves the definition of sequences without giving explicit formulae for their terms. The second involves proving that computer programs are correct.

Exercises

1. Find a formula for the sum of the first n even positive integers.
2. Use mathematical induction to prove the formula that you found in Exercise 1.
3. Use mathematical induction to prove that $3 + 3 \cdot 5 + 3 \cdot 5^2 + \cdots + 3 \cdot 5^n = 3(5^{n+1} - 1)/4$ whenever n is a nonnegative integer.
4. Use mathematical induction to prove that $2 - 2 \cdot 7 + 2 \cdot 7^2 - \cdots + 2(-7)^n = (1 - (-7)^{n+1})/4$ whenever n is a nonnegative integer.
5. Find a formula for

$$\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \cdots + \frac{1}{2^n}$$

by examining the values of this expression for small values of n. Use mathematical induction to prove your result.

6. Find a formula for

$$\frac{1}{1\cdot 2}+\frac{1}{2\cdot 3}+\cdots+\frac{1}{n(n+1)}$$

by examining the values of this expression for small values of n. Use mathematical induction to prove your result.

7. Show that $1^2+2^2+\cdots+n^2=n(n+1)(2n+1)/6$ whenever n is a positive integer.

8. Show that $1^3+2^3+\cdots+n^3=[n(n+1)/2]^2$ whenever n is a positive integer.

9. Prove that $1^2+3^2+5^2+\cdots+(2n+1)^2=(n+1)(2n+1)(2n+3)/3$ whenever n is a nonnegative integer.

10. Prove that $1\cdot 1!+2\cdot 2!+\cdots+n\cdot n!=(n+1)!-1$ whenever n is a positive integer.

***11.** Show by mathematical induction that if $h>-1$, then $1+nh\le(1+h)^n$ for all nonnegative integers n. This is called **Bernoulli's inequality.**

12. Prove that $3^n<n!$ whenever n is a positive integer greater than 6.

13. Show that $2^n>n^2$ whenever n is an integer greater than 4.

14. Use mathematical induction to prove that $n!<n^n$ whenever n is a positive integer greater than 1.

15. Prove using mathematical induction that

$$1\cdot 2+2\cdot 3+\cdots+n(n+1)$$
$$=n(n+1)(n+2)/3$$

whenever n is a positive integer.

16. Use mathematical induction to prove that

$$1\cdot 2\cdot 3+2\cdot 3\cdot 4+\cdots+n(n+1)(n+2)$$
$$=n(n+1)(n+2)(n+3)/4.$$

17. Show that $1^2-2^2+3^2-\cdots+(-1)^{n-1}n^2=(-1)^{n-1}n(n+1)/2$ whenever n is a positive integer.

18. Prove that

$$1+\frac{1}{4}+\frac{1}{9}+\cdots+\frac{1}{n^2}<2-\frac{1}{n}$$

whenever n is a positive integer greater than 1.

19. Show that any postage that is a positive integer number of cents greater than 7 cents can be formed using just 3-cent stamps and 5-cent stamps.

20. Use mathematical induction to show that 3 divides n^3+2n whenever n is a nonnegative integer.

21. Use mathematical induction to show that 5 divides n^5-n whenever n is a nonnegative integer.

22. Use mathematical induction to show that 6 divides n^3-n whenever n is a nonnegative integer.

***23.** Use mathematical induction to show that n^2-1 is divisible by 8 whenever n is an odd positive integer.

24. Use mathematical induction to show that $n^2-7n+12$ is nonnegative whenever n is an integer greater than 3.

25. Use mathematical induction to prove that a set with n elements has $n(n-1)/2$ subsets containing exactly two elements whenever n is an integer greater than or equal to 2.

***26.** Use mathematical induction to prove that a set with n elements has $n(n-1)(n-2)/6$ subsets containing exactly three elements whenever n is an integer greater than or equal to 3.

27. Use mathematical induction to prove that $\sum_{j=1}^{n} j^4=n(n+1)(2n+1)(3n^2+3n-1)/30$ whenever n is a positive integer.

28. For which nonnegative integers n is $n^2\le n!$? Prove your answer using mathematical induction.

29. For which nonnegative integers n is $2n+3\le 2^n$? Prove your answer using mathematical induction.

30. Use mathematical induction to show that $1/(2n)\le[1\cdot 3\cdot 5\cdot\cdots\cdot(2n-1)]/(2\cdot 4\cdot\cdots\cdot 2n)$ whenever n is a positive integer.

31. a) Determine which amounts of postage can be formed using just 5-cent and 6-cent stamps.
   b) Prove your answer to (a) using the principle of mathematical induction.
   c) Prove your answer to (a) using the second principle of mathematical induction.

32. Which amounts of money can be formed using just dimes and quarters? Prove your answer using a form of mathematical induction.

33. An automatic teller machine has only \$20 bills and \$50 bills. Which amounts of money can the machine dispense, assuming the machine has a limitless supply of these two denominations of bills? Prove your answer using a form of mathematical induction.

34. Use mathematical induction to prove that $\sum_{k=1}^{n} k2^k=(n-1)2^{n-1}+2$.

35. Show that

$$\sum_{\{a_1,\ldots,a_k\}\subseteq\{1,2,\ldots,n\}}\frac{1}{a_1a_2\cdots a_k}=n.$$

(Here the sum is over all nonempty subsets of the set of the n smallest positive integers.)

36. Use mathematical induction to show that given a set of $n+1$ positive integers, none exceeding $2n$, there is at least one integer in this set that divides another integer in the set.

37. (Calculus required) Use mathematical induction to prove that the derivative of $f(x)=x^n$ equals nx^{n-1} whenever n is a positive integer. (For the inductive step, use the product rule for derivatives.)

38. Suppose that

$$\mathbf{A}=\begin{bmatrix} a & 0\\ 0 & b\end{bmatrix}$$

where a and b are real numbers. Show that

$$\mathbf{A}^n = \begin{bmatrix} a^n & 0 \\ 0 & b^n \end{bmatrix}$$

for every positive integer n.

39. Suppose that **A** and **B** are square matrices with the property $\mathbf{AB} = \mathbf{BA}$. Show that $\mathbf{AB}^n = \mathbf{B}^n\mathbf{A}$ for every positive integer n.

40. Suppose that m is a positive integer. Use mathematical induction to prove that if a and b are integers with $a \equiv b \pmod{m}$, then $a^k \equiv b^k \pmod{m}$ whenever k is a nonnegative integer.

41. Use mathematical induction to show that if $A_1, A_2, \ldots, A_n$ and B are sets, then

$$(A_1 \cup A_2 \cup \cdots \cup A_n) \cap B = (A_1 \cap B) \cup (A_2 \cap B) \cup \cdots \cup (A_n \cap B).$$

42. Prove that if $A_1, A_2, \ldots, A_n$ and $B_1, B_2, \ldots, B_n$ are sets such that $A_k \subseteq B_k$ for $k = 1, 2, \ldots, n$, then

a) $\displaystyle\bigcup_{k=1}^{n} A_k \subseteq \bigcup_{k=1}^{n} B_k$. **b)** $\displaystyle\bigcap_{k=1}^{n} A_k \subseteq \bigcap_{k=1}^{n} B_k$.

43. Use mathematical induction to prove that if $A_1, A_2, \ldots, A_n$ are subsets of a universal set U, then

$$\overline{\bigcup_{k=1}^{n} A_k} = \bigcap_{k=1}^{n} \overline{A_k}.$$

44. Use mathematical induction to show that $\neg(p_1 \vee p_2 \vee \cdots \vee p_n)$ is equivalent to $\neg p_1 \wedge \neg p_2 \wedge \cdots \wedge \neg p_n$ whenever $p_1, p_2, \ldots, p_n$ are propositions.

***45.** Show that

$$[(p_1 \to p_2) \wedge (p_2 \to p_3) \wedge \cdots \wedge (p_{n-1} \to p_n)] \to [(p_1 \wedge p_2 \wedge \cdots \wedge p_{n-1}) \to p_n]$$

is a tautology whenever $p_1, p_2, \ldots, p_n$ are propositions.

46. Use the formula for the sum of the terms of a geometric progression to evaluate the following sums.

a) $4 + 4 \cdot 3 + 4 \cdot 3^2 + \cdots + 4 \cdot 3^8$

b) $3 + 3 \cdot 2^2 + 3 \cdot 2^4 + \cdots + 3 \cdot 2^{10}$

c) $1 - 2 + 2^2 - 2^3 + \cdots + (-1)^n 2^n$

47. What is wrong with the following "proof" that all horses are the same color?

> Let $P(n)$ be the proposition that all the horses in a set of n horses are the same color. Clearly, $P(1)$ is true. Now assume that $P(n)$ is true, so that all the horses in any set of n horses are the same color. Consider any $n + 1$ horses; number these as horses $1, 2, 3, \ldots, n, n + 1$. Now the first n of these horses all must have the same color, and the last n of these must also have the same color. Since the set of the first n horses and the set of the last n horses overlap, all $n + 1$ must be the same color. This shows that $P(n + 1)$ is true and finishes the proof by induction.

***48.** Find the flaw with the following "proof" that $a^n = 1$ for all nonnegative integers n, whenever a is a nonzero real number.

BASIS STEP: $a^0 = 1$ is true by the definition of a^0.

INDUCTIVE STEP: Assume that $a^k = 1$ for all nonnegative integers k with $k \le n$. Then note that

$$a^{n+1} = \frac{a^n \cdot a^n}{a^{n-1}} = \frac{1 \cdot 1}{1} = 1.$$

***49.** Show that the second form of mathematical induction is a valid method of proof by showing that it follows from the well-ordering property.

***50.** Show that the following form of mathematical induction is a valid method to prove that $P(n)$ is true for all positive integers n.

BASIS STEP: $P(1)$ and $P(2)$ are true.

INDUCTIVE STEP: For each positive integer n, if $P(n)$ and $P(n + 1)$ are both true, then $P(n + 2)$ is true.

In Exercises 51 and 52, H_n denotes the nth harmonic number.

***51.** Use mathematical induction to show that $H_{2^n} \le 1 + n$ whenever n is a nonnegative integer.

***52.** Use mathematical induction to prove that

$$H_1 + H_2 + \cdots + H_n = (n + 1)H_n - n.$$

***53.** Prove that

$$1 + \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{3}} + \cdots + \frac{1}{\sqrt{n}} > 2(\sqrt{n + 1} - 1).$$

***54.** Show that n lines separate the plane into $(n^2 + n + 2)/2$ regions if no two of these lines are parallel and no three pass through a common point.

****55.** Let $a_1, a_2, \ldots, a_n$ be positive real numbers. The **arithmetic mean** of these numbers is defined by

$$A = (a_1 + a_2 + \cdots + a_n)/n,$$

and the **geometric mean** of these numbers is defined by

$$G = (a_1 a_2 \cdots a_n)^{1/n}.$$

Use mathematical induction to prove that $A \ge G$.

***56.** Use mathematical induction to show that 21 divides $4^{n+1} + 5^{2n-1}$ whenever n is a positive integer.

57. Use mathematical induction to prove Lemma 2 of Section 2.5, which states that if p is a prime and $p \mid a_1 a_2 \cdots a_n$, where a_i is an integer for $i = 1, 2, 3, \ldots, n$, then $p \mid a_i$ for some integer i.

***58.** The well-ordering property can be used to show that there is a unique greatest common divisor of two positive integers. Let a and b be positive integers, and let S be the set of positive integers of the form $as + bt$, where s and t are integers.

a) Show that S is nonempty.

b) Use the well-ordering property to show that S has a smallest element c.

c) Show that if d is a common divisor of a and b, then d is a divisor of c.

d) Show that $c \mid a$ and $c \mid b$. (*Hint:* First, assume that $c \nmid a$. Then $a = qc + r$, where $0 < r < c$. Show that $r \in S$, contradicting the choice of c.)

e) Conclude from (c) and (d) that the greatest common divisor of a and b exists. Finish the proof by showing that this greatest common divisor of two positive integers is unique.

***59.** Show that if $a_1, a_2, \ldots, a_n$ are n distinct real numbers, exactly $n - 1$ multiplications are used to compute the product of these n numbers no matter how parentheses are inserted into their product. (*Hint:* Use the second principle of mathematical induction and consider the last multiplication.)

60. Construct a tiling using L-shaped pieces of the 4×4 chessboard with the square in the upper left corner removed.

61. Construct a tiling using L-shaped pieces of the 8×8 chessboard with the square in the upper left corner removed.

62. Prove or disprove that all chessboards of the following shapes can be completely covered using L-shaped pieces whenever n is a positive integer.

a) 3×2^n **b)** 6×2^n
c) $3^n \times 3^n$ **d)** $6^n \times 6^n$

***63.** Show that a three-dimensional $2^n \times 2^n \times 2^n$ chessboard with one $1 \times 1 \times 1$ cube missing can be completely covered by $2 \times 2 \times 2$ cubes with one $1 \times 1 \times 1$ cube removed.

***64.** Show that an $n \times n$ chessboard with one square removed can be completely covered using L-shaped pieces if $n > 5$, n is odd, and n is not divisible by 3.

65. Show that a 5×5 chessboard with a corner square removed can be tiled using L-shaped pieces.

***66.** Find a 5×5 chessboard with a square removed that cannot be tiled using L-shaped pieces. Prove that such a tiling does not exist for this board.

67. Let a be an integer and d be a positive integer. Show that the integers q and r with $a = dq + r$ and $0 \le r < d$, which were shown to exist in Example 1, are unique.

☞ **68.** Use the principle of mathematical induction to show that $P(n)$ is true for $n = k, k+1, k+2, \ldots$, where k is an integer, if $P(k)$ is true and the implication $P(n) \rightarrow P(n+1)$ is true for all positive integers n with $n \ge k$.

****69.** Can you use the well-ordering property to prove the following statement? "Every positive integer can be described using no more than 15 English words"?

3.3

Recursive Definitions

INTRODUCTION

Sometimes it is difficult to define an object explicitly. However, it may be easy to define this object in terms of itself. This process is called **recursion.** For instance, the picture shown in Figure 1 is produced recursively. First, an original picture is given. Then a process of successively superimposing centered smaller pictures on top of the previous pictures is carried out.

We can use recursion to define sequences, functions, and sets. In previous discussions, we specified the terms of a sequence using an explicit formula. For instance, the sequence of powers of 2 is given by $a_n = 2^n$ for $n = 0, 1, 2, \ldots$. However, this sequence can also be defined by giving the first term of the sequence, namely, $a_0 = 1$, and a rule for finding a term of the sequence from the previous one, namely, $a_{n+1} = 2a_n$ for $n = 0, 1, 2, \ldots$.

RECURSIVELY DEFINED FUNCTIONS

To define a function with the set of nonnegative integers as its domain,

1. *Specify the value of the function at zero.*
2. *Give a rule for finding its value as an integer from its values at smaller integers.*

Such a definition is called a **recursive** or **inductive definition.**

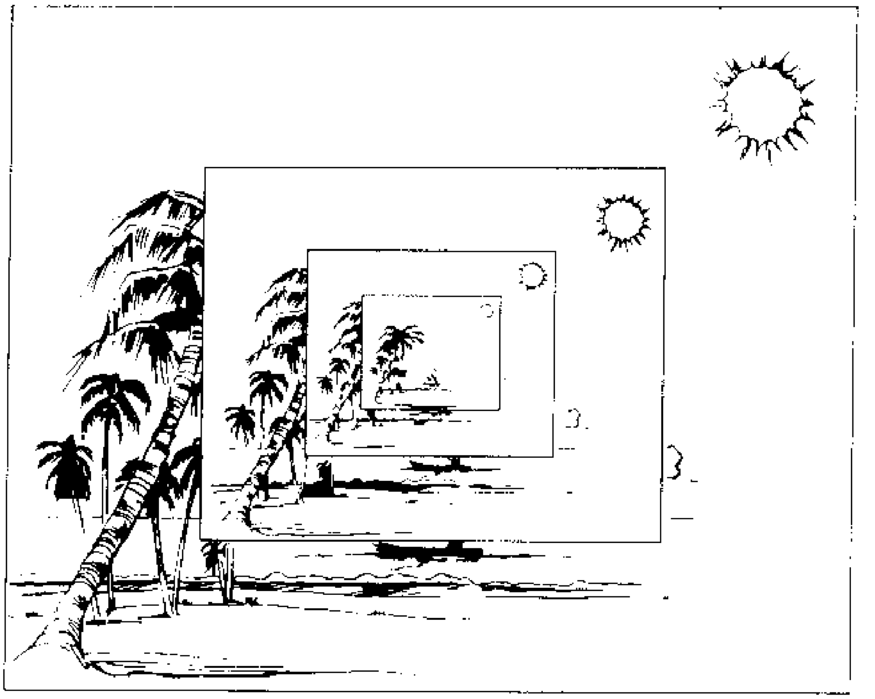


FIGURE 1 A Recursively Defined Picture.

EXAMPLE 1 Suppose that f is defined recursively by

$$f(0) = 3,$$
$$f(n+1) = 2f(n) + 3.$$

Find $f(1)$, $f(2)$, $f(3)$, and $f(4)$.

Solution: From the recursive definition it follows that

$$f(1) = 2f(0) + 3 = 2 \cdot 3 + 3 = 9,$$
$$f(2) = 2f(1) + 3 = 2 \cdot 9 + 3 = 21,$$
$$f(3) = 2f(2) + 3 = 2 \cdot 21 + 3 = 45,$$
$$f(4) = 2f(3) + 3 = 2 \cdot 45 + 3 = 93.$$

■

Many functions can be studied using their recursive definitions. The factorial function is one such example.

EXAMPLE 2 Give an inductive definition of the factorial function $F(n) = n!$.

Solution: We can define the factorial function by specifying the initial value of this function, namely, $F(0) = 1$, and giving a rule for finding $F(n+1)$ from $F(n)$. This is obtained by noting that $(n+1)!$ is computed from $n!$ by multiplying by $n+1$. Hence, the desired rule is

$$F(n+1) = (n+1)F(n). \qquad \blacksquare$$

To determine a value of the factorial function, such as $F(5) = 5!$, from the recursive definition found in Example 2, it is necessary to use the rule that shows how to express $F(n+1)$ in terms of $F(n)$ several times:

$$\begin{aligned} F(5) &= 5F(4) = 5\cdot 4F(3) = 5\cdot 4\cdot 3F(2) = 5\cdot 4\cdot 3\cdot 2F(1) \\ &= 5\cdot 4\cdot 3\cdot 2\cdot 1\cdot F(0) = 5\cdot 4\cdot 3\cdot 2\cdot 1\cdot 1 = 120. \end{aligned}$$

Once $F(0)$ is the only value of the function that occurs, no more reductions are necessary. The only thing left to do is to insert the value of $F(0)$ into the formula.

Recursively defined functions are well-defined. This is a consequence of the principle of mathematical induction. (See Exercise 44 at the end of this section.) Additional examples of recursive definitions are given in the following examples.

EXAMPLE 3 Give a recursive definition of a^n where a is a nonzero real number and n is a nonnegative integer.

Solution: The recursive definition contains two parts. First a^0 is specified, namely, $a^0 = 1$. Then the rule for finding a^{n+1} from a^n, namely, $a^{n+1} = a\cdot a^n$, for $n = 0, 1, 2, 3, \ldots$, is given. These two equations uniquely define a^n for all nonnegative integers n. ■

EXAMPLE 4 Give a recursive definition of

$$\sum_{k=0}^{n} a_k.$$

Solution: The first part of the recursive definition is

$$\sum_{k=0}^{0} a_k = a_0.$$

The second part is

$$\sum_{k=0}^{n+1} a_k = \left(\sum_{k=0}^{n} a_k\right) + a_{n+1}. \qquad \blacksquare$$

In some recursive definitions of functions, the values of the function at the first k positive integers are specified, and a rule is given for determining the value of the function at larger integers from its values at some or all of the preceding k integers. That such definitions produce well-defined functions follows from the second principle of mathematical induction (see Exercise 45 at the end of this section).

EXAMPLE 5 The *Fibonacci numbers*, $f_0, f_1, f_2, \ldots$, are defined by the equations $f_0 = 0$, $f_1 = 1$, and

web

$$f_n = f_{n-1} + f_{n-2}$$

for $n = 2, 3, 4, \ldots$. What are the Fibonacci numbers f_2, f_3, f_4, f_5, f_6?

Solution: Since the first part of the definition states that $f_0 = 0$ and $f_1 = 1$, it follows from the second part of the definition that

$$\begin{aligned}
f_2 &= f_1 + f_0 = 1 + 0 = 1,\\
f_3 &= f_2 + f_1 = 1 + 1 = 2,\\
f_4 &= f_3 + f_2 = 2 + 1 = 3,\\
f_5 &= f_4 + f_3 = 3 + 2 = 5,\\
f_6 &= f_5 + f_4 = 5 + 3 = 8.
\end{aligned}$$

■

We can use the recursive definition of the Fibonacci numbers to prove many properties of these numbers. We give one such property in the next example.

EXAMPLE 6 Show that $f_n > \alpha^{n-2}$, where $\alpha = (1 + \sqrt{5})/2$, whenever $n \geq 3$.

Solution: We can use the second principle of mathematical induction to prove this inequality. Let $P(n)$ be the statement $f_n > \alpha^{n-2}$. We want to show that $P(n)$ is true whenever n is an integer greater than or equal to 3.

First, note that

$$\alpha < 2 = f_3, \qquad \alpha^2 = (3 + \sqrt{5})/2 < 3 = f_4,$$

so that $P(3)$ and $P(4)$ are true. Now assume that $P(k)$ is true, namely, that $f_k > \alpha^{k-2}$, for all integers k with $3 \leq k \leq n$, where $n \geq 4$. We must show that $P(n+1)$ is true, that is, that $f_{n+1} > \alpha^{n-1}$. Since α is a solution of $x^2 - x - 1 = 0$ (as the quadratic formula shows), it follows that $\alpha^2 = \alpha + 1$. Therefore,

$$\alpha^{n-1} = \alpha^2 \cdot \alpha^{n-3} = (\alpha + 1)\alpha^{n-3} = \alpha \cdot \alpha^{n-3} + 1 \cdot \alpha^{n-3} = \alpha^{n-2} + \alpha^{n-3}.$$

By the inductive hypothesis, if $n \geq 5$, it follows that

$$f_{n-1} > \alpha^{n-3}, \qquad f_n > \alpha^{n-2}.$$

Therefore, we have

$$f_{n+1} = f_n + f_{n-1} > \alpha^{n-2} + \alpha^{n-3} = \alpha^{n-1}.$$

It follows that $P(n+1)$ is true. This completes the proof. ■

---

web

Fibonacci (c. 1180–1228). Fibonacci (short for *filius Bonacci*, or "son of Bonacci") was also known as Leonardo of Pisa. He was born in the Italian commercial center of Pisa. Fibonacci was a merchant who traveled extensively throughout the Mideast, where he came into contact with Arabian mathematics. In his book *Liber Abaci*, Fibonacci introduced the European world to Arabic notation for numerals and algorithms for arithmetic. It was in this book that his famous rabbit problem (described in Section 5.1) appeared. Fibonacci also wrote books on geometry and trigonometry and on Diophantine equations, which involve finding integer solutions to equations.

Remark: The inductive step shows that whenever $n \geq 4$, $P(n+1)$ follows from the assumption that $P(k)$ is true for $3 \leq k \leq n$. Hence, the inductive step does *not* show that $P(3) \to P(4)$. Therefore, we had to show that $P(4)$ is true separately.

We can now show that the Euclidean algorithm uses $O(\log b)$ divisions to find the greatest common divisor of the positive integers a and b, where $a \geq b$.

THEOREM 1 **LAMÉ'S THEOREM Let a and b be positive integers with $a \geq b$. Then the number of divisions used by the Euclidean algorithm to find $\gcd(a, b)$ is less than or equal to five times the number of decimal digits in b.**

Proof: Recall that when the Euclidean algorithm is applied to find $\gcd(a, b)$ with $a \geq b$, the following sequence of equations (where $a = r_0$ and $b = r_1$) is obtained.

$$\begin{array}{ll}
r_0 = r_1 q_1 + r_2 & 0 \leq r_2 < r_1 \\
r_1 = r_2 q_2 + r_3 & 0 \leq r_3 < r_2 \\
\quad \cdot & \\
\quad \cdot & \\
\quad \cdot & \\
r_{n-2} = r_{n-1} q_{n-1} + r_n & 0 \leq r_n < r_{n-1} \\
r_{n-1} = r_n q_n. &
\end{array}$$

Here n divisions have been used to find $r_n = \gcd(a, b)$. Note that the quotients $q_1, q_2, \ldots, q_{n-1}$ are all at least 1. Moreover, $q_n \geq 2$, since $r_n < r_{n-1}$. This implies that

$$\begin{aligned}
r_n &\geq 1 = f_2, \\
r_{n-1} &\geq 2r_n \geq 2f_2 = f_3, \\
r_{n-2} &\geq r_{n-1} + r_n \geq f_3 + f_2 = f_4, \\
&\ \ \cdot \\
&\ \ \cdot \\
&\ \ \cdot \\
r_2 &\geq r_3 + r_4 \geq f_{n-1} + f_{n-2} = f_n, \\
b = r_1 &\geq r_2 + r_3 \geq f_n + f_{n-1} = f_{n-1}.
\end{aligned}$$

It follows that if n divisions are used by the Euclidean algorithm to find $\gcd(a, b)$ with $a \geq b$, then $b \geq f_{n+1}$. From Example 6 we know that $f_{n+1} > \alpha^{n-1}$ for $n > 2$, where $\alpha = (1 + \sqrt{5})/2$. Therefore, it follows that $b > \alpha^{n-1}$. Furthermore, since $\log_{10} \alpha \sim 0.208 > 1/5$, we see that

$$\log_{10} b > (n-1) \log_{10} \alpha > (n-1)/5.$$

Hence, $n - 1 < 5 \cdot \log_{10} b$. Now suppose that b has k decimal digits. Then $b < 10^k$ and $\log_{10} b < k$. It follows that $n - 1 < 5k$, and since k is an integer, it follows that $n \leq 5k$. This finishes the proof. □

Since the number of decimal digits in b, which equals $\lfloor \log_{10} b \rfloor + 1$, is less than or equal to $\log_{10} b + 1$, Theorem 1 tells us that the number of divisions required to find $\gcd(a, b)$ with $a > b$ is less than or equal to $5(\log_{10} b + 1)$. Since $5(\log_{10} b + 1)$

is $O(\log b)$, we see that $O(\log b)$ divisions are used by the Euclidean algorithm to find $\gcd(a, b)$ whenever $a > b$.

RECURSIVELY DEFINED SETS

Recursive definitions are often used to define sets. When this is done, an initial collection of elements is given. Then the rules used to construct elements of the set from other elements already known to be in the set are given. Sets described in this way are well-defined, and theorems about them can be proved using their recursive definitions. Some examples of recursive definitions of sets follow.

EXAMPLE 7 Let S be defined recursively by

$3 \in S$;
$x + y \in S$ if $x \in S$ and $y \in S$.

Show that S is the set of positive integers divisible by 3. (Note that implicit in this definition is the assumption that nothing belongs to S unless it can be generated using the two statements in the recursive definition of S.)

Solution: Let A be the set of all positive integers divisible by 3. To prove that $A = S$, we must show that A is a subset of S and that S is a subset of A. To prove that A is a subset of S, we must show that every positive integer divisible by 3 is in S. We will use mathematical induction to prove this.

Let $P(n)$ be the statement that $3n$ belongs to S. The basis step holds since, by the first part of the recursive definition of S, $3 \times 1 = 3$ is in S. To establish the inductive step, assume that $P(n)$ is true, namely, that $3n$ is in S. Since $3n$ is in S and since 3 is in S, it follows from the second part of the recursive definition of S that $3n + 3 = 3(n + 1)$ is also in S.

To prove that S is a subset of A, we use the recursive definition of S. First, the basis step of the definition specifies that 3 is in S. Since $3 = 3 \times 1$, all elements specified to be in S in this step are divisible by 3. To finish the proof, we must show that all integers in S generated using the second part of the recursive definition are in A. This consists of showing that $x + y$ is in A whenever x and y are elements of S also assumed to be in A. Now if x and y are both in A, it follows that $3 \mid x$ and $3 \mid y$. By Theorem 1 of Section 2.3, it follows that $3 \mid x + y$, completing the proof. ∎

---

web **Gabriel Lamé (1795–1870).** Gabriel Lamé entered the École Polytechnique in 1813, graduating in 1817. He continued his education at the École des Mines, graduating in 1820.

In 1820 Lamé went to Russia, where he was appointed director of the Schools of Highways and Transportation in St. Petersburg. Not only did he teach, but he also planned roads and bridges while in Russia. He returned to Paris in 1832, where he helped found an engineering firm. However, he soon left the firm, accepting the chair of physics at the École Polytechnique, which he held until 1844. While holding this position, he was active outside academia as an engineering consultant, serving as chief engineer of mines and participating in the building of railways.

Lamé contributed original work to number theory, applied mathematics, and thermodynamics. His best-known work involves the introduction of curvilinear coordinates. His work on number theory includes proving Fermat's Last Theorem for $n = 7$, as well as providing the upper bound for the number of divisions used by the Euclidean algorithm given in this text.

In the opinion of Gauss, one of the most important mathematicians of all time, Lamé was the foremost French mathematician of his time. However, French mathematicians considered him too practical, whereas French scientists considered him too theoretical.

The recursive definition of a set in Example 7 is typical. First, an initial set of elements is given. Second, a rule is provided for generating new elements from those already known to be in the set. Implicit in the definition is that no element belongs to the set unless it is listed in the initial set of elements or it can be generated using the rule given for constructing new elements.

One of the most common uses of recursive definitions for sets is to define **well-formed formulae** in various systems. This is illustrated in the following examples.

EXAMPLE 8 The well-formed formulae of variables, numerals, and operators from $\{+, -, *, /, \uparrow\}$ (where $*$ denotes multiplication and $\uparrow$ denotes exponentiation) are defined by

x is a well-formed formula if x is a numeral or a variable;
$(f + g)$, $(f - g)$, $(f * g)$, (f/g), and $(f \uparrow g)$ are well-formed formulae if f and g are.

For instance, from this definition, since x and 3 are well-formed formulae, $(x + 3)$, $(x - 3)$, $(x * 3)$, $(x/3)$, and $(x \uparrow 3)$ are well-formed formulae. Continuing, since y also is a well-formed formula, so are $((x + 3) + y)$, $(y - (x * 3))$, and so on. (Note that $(3/0)$ is a well-formed formula, since only syntax matters here.) ■

EXAMPLE 9 The well-formed formulae for compound propositions involving **T**, **F**, propositional variables, and operators $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ are defined by

T, **F**, and p, where p is a propositional variable, are well-formed formulae; $(\neg p)$, $(p \vee q)$, $(p \wedge q)$, $(p \rightarrow q)$, $(p \leftrightarrow q)$ are well-formed formulae if p and q are well-formed formulae.

For example, if p, q, and r are propositional variables, then repeatedly using the recursive definition shows that $(p \vee q)$, $(r \wedge \mathbf{T})$, and $((p \vee q) \rightarrow (r \wedge \mathbf{T}))$ are well-formed formulae. ■

Recursive definitions are often used in the study of strings. Recall from Chapter 1 that a **string** over an alphabet Σ is a finite sequence of symbols from Σ. The set of strings over Σ is denoted by Σ^*. Two strings can be combined via the operation of **concatenation.** The concatenation of the strings x and y, denoted by xy, is the string x followed by the string y. For instance, the concatenation of x = *abra* and y = *cadabra* is xy = *abracadabra*. The following recursive definition is often used when results about strings are proved.

EXAMPLE 10 A Recursive Definition of the Set of Strings The set Σ^* of strings over the alphabet Σ can be defined recursively by $\lambda \in \Sigma^*$, where λ is the *empty string* containing no symbols, and $wx \in \Sigma^*$ whenever $w \in \Sigma^*$ and $x \in \Sigma$.

The first part of this definition says that the empty string belongs to Σ^*. The second part states that new strings are produced by concatenating strings in Σ^* with symbols from Σ. ■

The **length** of a string, which is the number of symbols in the string, can also be defined recursively.

EXAMPLE 11 Give a recursive definition of $l(w)$, the length of the string w.

Solution: The length of a string can be defined by

$$l(\lambda) = 0;$$

$$l(wx) = l(w) + 1 \text{ if } w \in \Sigma^* \text{ and } x \in \Sigma. \quad \blacksquare$$

The following example illustrates how the recursive definition of strings can be used in proofs.

EXAMPLE 12 Use mathematical induction to prove that $l(xy) = l(x) + l(y)$, where x and y belong to Σ^*, the set of strings over the alphabet Σ.

Solution: We will base our proof on the recursive definition of the set Σ^* given in Example 10. Let $P(y)$ be the statement that $l(xy) = l(x) + l(y)$ whenever x belongs to Σ^*.

BASIS STEP: To complete the basis step, we must show that $P(\lambda)$ is true. That is, we must show that $l(x\lambda) = l(x)+l(\lambda)$ for all $x \in \Sigma^*$. Since $l(x\lambda) = l(x) = l(x)+0 = l(x) + l(\lambda)$ for every string x, it follows that $P(\lambda)$ is true.

INDUCTIVE STEP: To complete the inductive step, we assume that $P(y)$ is true and show that this implies that $P(ya)$ is true whenever $a \in \Sigma$. What we need to show is that $l(xya) = l(x)+l(ya)$ for every $a \in \Sigma$. To show this, note that by the recursive definition of $l(w)$ (given in Example 11), we have $l(xya) = l(xy)+1$ and $l(ya) = l(y)+1$. And, by the inductive hypothesis, $l(xy) = l(x) + l(y)$. We conclude that $l(xya) = l(x) + l(y) + 1 = l(x) + l(ya)$. $\blacksquare$

Exercises

1. Find $f(1)$, $f(2)$, $f(3)$, and $f(4)$ if $f(n)$ is defined recursively by $f(0) = 1$ and for $n = 0, 1, 2, \ldots$

a) $f(n+1) = f(n) + 2$.

b) $f(n+1) = 3f(n)$.

c) $f(n+1) = 2^{f(n)}$.

d) $f(n+1) = f(n)^2 + f(n) + 1$.

2. Find $f(1)$, $f(2)$, $f(3)$, $f(4)$, and $f(5)$ if $f(n)$ is defined recursively by $f(0) = 3$ and for $n = 0, 1, 2, \ldots$

a) $f(n+1) = -2f(n)$.

b) $f(n+1) = 3f(n) + 7$.

c) $f(n+1) = f(n)^2 - 2f(n) - 2$.

d) $f(n+1) = 3^{f(n)/3}$.

3. Find $f(2)$, $f(3)$, $f(4)$, and $f(5)$ if f is defined recursively by $f(0) = -1$, $f(1) = 2$ and for $n = 1, 2, \ldots$

a) $f(n+1) = f(n) + 3f(n-1)$.

b) $f(n+1) = f(n)^2 f(n-1)$.

c) $f(n+1) = 3f(n)^2 - 4f(n-1)^2$.

d) $f(n+1) = f(n-1)/f(n)$.

4. Find $f(2)$, $f(3)$, $f(4)$, and $f(5)$ if f is defined recursively by $f(0) = f(1) = 1$ and for $n = 1, 2, \ldots$

a) $f(n+1) = f(n) - f(n-1)$.

b) $f(n+1) = f(n)f(n-1)$.

c) $f(n+1) = f(n)^2 + f(n-1)^3$.

d) $f(n+1) = f(n)/f(n-1)$.

5. Give a recursive definition of the sequence $\{a_n\}$, $n = 1, 2, 3, \ldots$ if

a) $a_n = 6n$. **b)** $a_n = 2n + 1$.

c) $a_n = 10^n$. **d)** $a_n = 5$.

6. Give a recursive definition of the sequence $\{a_n\}$, $n = 1, 2, 3, \ldots$ if

a) $a_n = 4n - 2$. **b)** $a_n = 1 + (-1)^n$.

c) $a_n = n(n+1)$. **d)** $a_n = n^2$.

7. Let F be the function such that $F(n)$ is the sum of the first n positive integers. Give a recursive definition of $F(n)$.

8. Give a recursive definition of $S_m(n)$, the sum of the integer m and the nonnegative integer n.

9. Give a recursive definition of $P_m(n)$, the product of the integer m and the nonnegative integer n.

In Exercises 10–17 f_n is the nth Fibonacci number.

10. Prove that $f_1^2 + f_2^2 + \cdots + f_n^2 = f_n f_{n+1}$ whenever n is a positive integer.

11. Prove that $f_1 + f_3 + \cdots + f_{2n-1} = f_{2n}$ whenever n is a positive integer.

***12.** Show that $f_{n+1} f_{n-1} - f_n^2 = (-1)^n$ whenever n is a positive integer.

***13.** Show that $f_0 f_1 + f_1 f_2 + \cdots + f_{2n-1} f_{2n} = f_{2n}^2$ whenever n is a positive integer.

***14.** Show that $f_0 - f_1 + f_2 - \cdots - f_{2n-1} + f_{2n} = f_{2n-1} - 1$ whenever n is a positive integer.

15. Determine the number of divisions used by the Euclidean algorithm to find the greatest common divisor of the Fibonacci numbers f_n and f_{n+1} where n is a nonnegative integer. Verify your answer using mathematical induction.

16. Let

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

Show that

$$\mathbf{A}^n = \begin{bmatrix} f_{n+1} & f_n \\ f_n & f_{n-1} \end{bmatrix}$$

whenever n is a positive integer.

17. By taking determinants of both sides of the equation in Exercise 16, prove the identity given in Exercise 12. (This exercise depends on the notion of the determinant of a 2×2 matrix.)

***18.** Give a recursive definition of the functions max and min so that $\max(a_1, a_2, \ldots, a_n)$ and $\min(a_1, a_2, \ldots, a_n)$ are the maximum and minimum of the n numbers $a_1, a_2, \ldots, a_n$, respectively.

***19.** Let $a_1, a_2, \ldots, a_n$, and $b_1, b_2, \ldots, b_n$ be real numbers. Use the recursive definitions that you gave in Exercise 18 to prove the following.

a) $\max(-a_1, -a_2, \ldots, -a_n) = -\min(a_1, a_2, \ldots, a_n)$

b) $\max(a_1 + b_1, a_2 + b_2, \ldots, a_n + b_n)$
$\leq \max(a_1, a_2, \ldots, a_n) + \max(b_1, b_2, \ldots, b_n)$

c) $\min(a_1 + b_1, a_2 + b_2, \ldots, a_n + b_n)$
$\geq \min(a_1, a_2, \ldots, a_n) + \min(b_1, b_2, \ldots, b_n)$

20. Show that the set S defined by $1 \in S$ and $s + t \in S$ whenever $s \in S$ and $t \in S$ is the set of positive integers.

21. Give a recursive definition of the set of positive integers that are multiples of 5.

22. Give a recursive definition of

a) the set of odd positive integers.

b) the set of positive integer powers of 3.

c) the set of polynomials with integer coefficients.

23. Give a recursive definition of

a) the set of even integers.

b) the set of positive integers congruent to 2 modulo 3.

c) the set of positive integers not divisible by 5.

24. Show that any well-formed formula of numerals, variables, and operators from $\{+, -, *, /, \uparrow\}$ contains the same number of right and left parentheses.

25. Define well-formed formulae of sets, variables representing sets, and operators from $\{\bar{\ }, \cup, \cap, -\}$.

The **reversal** of a string is the string consisting of the symbols of the string in reverse order. The reversal of the string w is denoted by w^R.

26. Find the reversal of the following bit strings.

a) 0101

b) 1 1011

c) 1000 1001 0111

27. Give a recursive definition of the reversal of a string. (*Hint:* First define the reversal of the empty string. Then write a string w of length $n + 1$ as xy, where x is a string of length n, and express the reversal of w in terms of x^R and y.)

***28.** Give a recursive proof that $(w_1 w_2)^R = w_2^R w_1^R$.

29. Give a recursive definition of w^i where w is a string and i is a nonnegative integer. (Here w^i represents the concatenation of i copies of the string w.)

***30.** Give a recursive definition of the set of bit strings that are palindromes.

31. When does a string belong to the set A of bit strings defined recursively by

$$\lambda \in A$$

$$0x1 \in A \text{ if } x \in A,$$

where λ is the empty string?

***32.** Recursively define the set of bit strings that have more 0s than 1s.

33. Use Exercise 29 and mathematical induction to show that $l(w^i) = i \cdot l(w)$, where w is a string and i is a nonnegative integer.

***34.** Show that $(w^R)^i = (w^i)^R$ whenever w is a string and i is a nonnegative integer; that is, show that the ith power of the reversal of a string is the reversal of the ith power of the string.

***35.** A **partition** of a positive integer n is a way to write n as a sum of positive integers. For instance, $7 = 3 + 2 + 1 + 1$ is a partition of 7. Let P_m equal the number of different partitions of m, where the order of terms in the sum does not matter, and let $P_{m,n}$ be the number of different ways to express m as the sum of positive integers not exceeding n.

a) Show that $P_{m,m} = P_m$.

b) Show that the following recursive definition for $P_{m,n}$ is correct:

$$P_{m,n} = \begin{cases} 1 & \text{if } m = 1 \\ 1 & \text{if } n = 1 \\ P_{m,m} & \text{if } m < n \\ 1 + P_{m,m-1} & \text{if } m = n > 1 \\ P_{m,n-1} + P_{m-n,n} & \text{if } m > n > 1 \end{cases}$$

c) Find the number of partitions of 5 and of 6 using this recursive definition.

Consider the following inductive definition of a version of **Ackermann's function.** This function was named after Wilhelm Ackermann, a German mathematician who was a student of the great mathematician David Hilbert. Ackermann's function plays an important role in the theory of recursive functions and in the study of the complexity of certain algorithms involving set unions. (There are several different variants of this function. All are called Ackermann's function and have similar properties even though their values do not always agree.) web

$$A(m, n) = \begin{cases} 2n & \text{if } m = 0 \\ 0 & \text{if } m \geq 1 \text{ and } n = 0 \\ 2 & \text{if } m \geq 1 \text{ and } n = 1 \\ A(m-1, A(m, n-1)) & \\ & \text{if } m \geq 1 \text{ and } n \geq 2 \end{cases}$$

Exercises 36 to 43 involve this version of Ackermann's function.

36. Find the following values of Ackermann's function.
a) $A(1, 0)$ **b)** $A(0, 1)$
c) $A(1, 1)$ **d)** $A(2, 2)$

37. Show that $A(m, 2) = 4$ whenever $m \geq 1$.

38. Show that $A(1, n) = 2^n$ whenever $n \geq 1$.

39. Find the following values of Ackermann's function.
a) $A(2, 3)$ ***b)** $A(3, 3)$

***40.** Find $A(3, 4)$.

****41.** Prove that $A(m, n + 1) > A(m, n)$ whenever m and n are nonnegative integers.

***42.** Prove that $A(m + 1, n) \geq A(m, n)$ whenever m and n are nonnegative integers.

43. Prove that $A(i, j) \geq j$ whenever i and j are nonnegative integers.

44. Use mathematical induction to prove that a function F defined by specifying $F(0)$ and a rule for obtaining $F(n + 1)$ from $F(n)$ is well-defined.

45. Use the second principle of mathematical induction to prove that a function F defined by specifying $F(0)$ and a rule for obtaining $F(n + 1)$ from the values $F(k)$ for $k = 0, 1, 2, \ldots, n$ is well-defined.

46. Show that each of the following proposed recursive definitions of a function on the set of positive integers does not produce a well-defined function.
a) $F(n) = 1 + F(\lfloor n/2 \rfloor)$ for $n \geq 1$ and $F(1) = 1$.
b) $F(n) = 1 + F(n - 3)$ for $n \geq 2$, $F(1) = 2$, and $F(2) = 3$.
c) $F(n) = 1 + F(n/2)$ for $n \geq 2$, $F(1) = 1$, and $F(2) = 2$.
d) $F(n) = 1 + F(n/2)$ if n is even and $n \geq 2$, $F(n) = 1 - F(n - 1)$ if n is odd, and $F(1) = 1$.
e) $F(n) = 1 + F(n/2)$ if n even and $n \geq 2$, $F(n) = F(3n - 1)$ if n is odd and $n \geq 3$, and $F(1) = 1$.

47. Show that each of the following proposed recursive definitions of a function on the set of positive integers does not produce a well-defined function.
a) $F(n) = 1 + F(\lfloor (n + 1)/2 \rfloor)$ for $n \geq 1$ and $F(1) = 1$.
b) $F(n) = 1 + F(n - 2)$ for $n \geq 2$ and $F(1) = 0$.
c) $F(n) = 1 + F(n/3)$ for $n \geq 3$, $F(1) = 1$, $F(2) = 2$, and $F(3) = 3$.
d) $F(n) = 1 + F(n/2)$ if n is even and $n \geq 2$, $F(n) = 1 + F(n - 2)$ if n is odd, and $F(1) = 1$.
e) $F(n) = 1 + F(F(n - 1))$ if $n \geq 2$ and $F(1) = 2$.

Exercises 48–50 deal with iterations of the logarithm function. Let $\log n$ denote the logarithm of n to the base 2, as usual. The function $\log^{(k)} n$ is defined recursively by

$$\log^{(k)} n = \begin{cases} n & \text{if } k = 0 \\ \log(\log^{(k-1)} n) & \text{if } \log^{(k-1)} n \text{ is defined} \\ & \text{and positive} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

The **iterated logarithm** is the function $\log^* n$ whose value at n is the smallest nonnegative integer k such that $\log^{(k)} n \leq 1$.

48. Find each of the following values:
a) $\log^{(2)} 16$ **b)** $\log^{(3)} 256$
c) $\log^{(3)} 2^{65536}$ **d)** $\log^{(4)} 2^{2^{65536}}$

49. Find the value of $\log^* n$ for each of the following values of n:
a) 2 **b)** 4 **c)** 8 **d)** 16
e) 256 **f)** 65536 **g)** 2^{2048}

50. Find the largest integer n such that $\log^* n = 5$. Determine the number of decimal digits in this number.

Exercises 51–53 deal with values of iterated functions. Suppose that $f(n)$ is a function from the set of real numbers, or positive real numbers, or some other set of real numbers, to the set of real numbers such that $f(n)$ is monotonically increasing [that is, $f(n) < f(m)$ when $n < m$] and $f(n) < n$ for all n in the domain of f.] The function $f^{(k)}(n)$ is defined recursively by

$$f^{(k)}(n) = \begin{cases} n & \text{if } k = 0 \\ f(f^{(k-1)}(n)) & \text{if } k > 0. \end{cases}$$

Furthermore, let c be a positive real number. The **iterated function** f_c^* is the number of iterations of f required to reduce its argument to c or less, so that $f_c^*(n)$ is the smallest nonnegative integer k such that $f^k(n) \leq c$.

51. Let $f(n) = n - a$, where a is a positive integer. Find a formula for $f^{(k)}(n)$. What is the value of $f_0^*(n)$ when n is a positive integer?

52. Let $f(n) = n/2$. Find a formula for $f^{(k)}(n)$. What is the value of $f_1^*(n)$ when n is a positive integer?

53. Let $f(n) = \sqrt{n}$. Find a formula for $f^{(k)}(n)$. What is the value of $f_2^*(n)$ when n is a positive integer?

Exercises 54–61 deal with some unusual sequences, informally called **self-generating sequences**, produced by simple recurrence relations or rules. In particular, Exercises 54–57 deal with the sequence $\{a(n)\}$ defined by $a(n) = n - a(a(n-1))$ for $n \geq 1$ and $a(0) = 0$. (This sequence, as well as those in Exercises 58 and 59, are defined in Douglas Hofstader's fascinating book *Gödel, Escher, Bach* ([Ho99]).

54. Find the first 10 terms of the sequence $\{a(n)\}$ defined in the preamble to this exercise.

***55.** Prove that this sequence is well-defined. That is, show that $a(n)$ is uniquely defined for all nonnegative integers n.

****56.** Prove that $a(n) = \lfloor (n+1)\mu \rfloor$ where $\mu = (-1 + \sqrt{5})/2$. [*Hint:* First show for all $n > 0$ that $(\mu n - \lfloor \mu n \rfloor) + (\mu^2 n - \lfloor \mu^2 n \rfloor) = 1$. Then show for all real numbers α with $0 \leq \alpha < 1$ and $\alpha \neq 1 - \mu$ that $\lfloor (1+\mu)(1-\alpha) \rfloor + \lfloor \alpha + \mu \rfloor = 1$, considering the cases $0 \leq \alpha < 1 - \mu$ and $1 - \mu < \alpha < 1$ separately.]

***57.** Use the formula from Exercise 56 to show that $a(n) = a(n-1)$ if $\mu n - \lfloor \mu n \rfloor < 1 - \mu$ and $a(n) = a(n-1) + 1$ otherwise.

58. Find the first 10 terms of each of the following self-generating sequences:

a) $a(n) = n - a(a(a(n-1)))$ for $n \geq 1$ with $a(0) = 0$.

b) $a(n) = n - a(a(a(a(n-1))))$ for $n \geq 1$ with $a(0) = 0$.

c) $a(n) = a(n - a(n-1)) + a(n - a(n-2))$ for $n \geq 3$ with $a(1) = 1$ and $a(2) = 1$.

59. Find the first 10 terms of both the sequences $m(n)$ and $f(n)$ defined by the following pair of interwoven recurrence relations: $m(n) = n - f(m(n-1))$, $f(n) = n - m(f(n-1))$ for $n \geq 1$ with $f(0) = 1$ and $m(0) = 0$.

web

Godfrey Harold Hardy (1877–1947). Hardy, born in Cranleigh, Surrey, England, was the older of two children of Isaac Hardy and Sophia Hall Hardy. His father was the geography and drawing master at the Cranleigh School and also gave singing lessons and played soccer. His mother gave piano lessons and helped run a boardinghouse for young students. Hardy's parents were devoted to their children's education. Hardy demonstrated his numerical ability at the early age of 2 when he began writing down numbers into the millions. He had a private mathematics tutor rather than attending regular classes at the Cranleigh School. He moved to Winchester College, a private high school, when he was 13 and was awarded a scholarship. He excelled in his studies and demonstrated a strong interest in mathematics. He entered Trinity College, Cambridge, in 1896 on a scholarship and won several prizes during his time there, graduating in 1899.

Hardy held the position of lecturer in mathematics at Trinity College at Cambridge University from 1906 to 1919, when he was appointed to the Sullivan chair of geometry at Oxford. He had become unhappy at Cambridge with the dismissal of the famous philosopher and mathematician Bertrand Russell from Trinity for antiwar activities and did not like a heavy load of administrative duties. In 1931 he returned to Cambridge as the Sadleirian professor of pure mathematics, where he remained until his retirement in 1942. He was a pure mathematician and held an elitist view of mathematics, hoping his research could never be applied. Ironically, he is perhaps best known as one of the developers of the Hardy–Weinberg law, which predicts patterns of inheritance. His work in this area appeared as a letter to the journal *Science* in which he used simple algebraic ideas to demonstrate errors in an article on genetics. Hardy worked primarily in number theory and function theory, working on such topics as the Riemann zeta function, Fourier series, and the distribution of primes. He made many important contributions to many important problems, such as Waring's problem about representing positive integers as the sum of kth powers and the problem of representing odd integers as the sum of three primes. Hardy is also remembered for his collaborations with John E. Littlewood, a colleague at Cambridge, with whom he wrote more than 100 papers and the famous Indian mathematical prodigy Srinivasa Ramanujan. His collaboration with Littlewood led to the joke that there were only three important English mathematicians at that time, Hardy, Littlewood, and Hardy–Littlewood, although some people thought that Hardy had invented a fictitious person, Littlewood, since Littlewood was seldom seen outside Cambridge. Hardy had the wisdom of recognizing Ramanujan's genius from unconventional but extremely creative writings Ramanujan sent him, while other mathematicians failed to see the genius. Hardy brought Ramanujan to Cambridge and collaborated on important joint papers, establishing new results on the number of partitions of an integer. Hardy was interested in mathematics education, and his book *A Course in Pure Mathematics* had a profound effect on undergraduate instruction in mathematics in the first half of the twentieth century. Hardy also wrote *A Mathematician's Apology* in which he gives his answer to the question whether it is worthwhile to devote one's life to the study of mathematics. It presents Hardy's view of what mathematics is and what a mathematician does.

Hardy had a strong interest in sports. He was an avid cricket fan and followed scores closely. One peculiar trait he had was that he did not like his picture taken (only five snapshots are known) and disliked mirrors, covering them with towels immediately upon entering a hotel room.

Golomb's self-generating sequence is the unique nondecreasing sequence of positive integers $a_1, a_2, a_3, \ldots$ which has the property that it contains exactly a_k occurrences of k for each positive integer k.

60. Find the first 20 terms of Golomb's self-generating sequence.

***61.** Show that if $f(n)$ is the largest integer m such that $a_m = n$, where a_m is the mth term of Golomb's self-generating sequence, then $f(n) = \sum_{k=1}^{n} a_k$ and $f(f(n)) = \sum_{k=1}^{n} ka_k$.

The set $\mathcal{L}$ of **logarithmico-exponential functions,** introduced by the famous British mathematician G. H. Hardy, is the smallest set of functions such that:

- the function $f(n) = \alpha$ belongs to $\mathcal{L}$, whenever α is a real number;
- the function $f(n) = n$ belongs to $\mathcal{L}$;
- if the functions $f(n)$ and $g(n)$ belong to $\mathcal{L}$, then $f(n) - g(n)$ belongs to $\mathcal{L}$;
- if the function $f(n)$ belongs to $\mathcal{L}$, then $e^{f(n)}$ belongs to $\mathcal{L}$;
- if $f(n)$ belongs to $\mathcal{L}$ and there exists an integer N such $f(n) > 0$ for $n \geq N$ (this means that f is called **eventually positive**), then $\ln f(n)$ belongs to $\mathcal{L}$, where $\ln x$ denotes the natural logarithm of x, as usual.

Hardy showed for every logarithmico-exponential function not identically zero that $f(n)$ is either eventually positive or eventually negative. He proved that if $f(n)$ and $g(n)$ belong to $\mathcal{L}$, then either $f(n)$ is $o(g(n))$, $g(n)$ is $o(f(n))$, or $f(n)$ and $g(n)$ are of the same order.

62. Show that if $f(n)$ and $g(n)$ belong to $\mathcal{L}$, then $f(n) + g(n)$ belong to $\mathcal{L}$.

***63.** Show that if $f(n)$ and $g(n)$ belong to $\mathcal{L}$ and are eventually positive, then $f(n)g(n)$ and $f(n)/g(n)$ belong to $\mathcal{L}$, and that this implies, using the fact that every logarithmico-exponential function is eventually positive, eventually negative, or identically zero, that the product and quotient of any two functions, not identically zero, in $\mathcal{L}$ are in $\mathcal{L}$.

64. Use Exercises 62 and 63 to show that every polynomial $f(n) = a_m n^m + a_{m-1} n^{m-1} + \cdots + a_0$ with real coefficients belongs to $\mathcal{L}$.

65. Show that if $f(n)$ belongs to $\mathcal{L}$ and is eventually positive, then $\sqrt{f(n)}$ belongs to $\mathcal{L}$.

66. Show that the function $e^{\sqrt{n \ln n \ln \ln n}}$ belongs to $\mathcal{L}$.

---

web

Srinivasa Ramanujan (1887–1920). The famous mathematical prodigy Ramanujan was born and raised in southern India near the city of Madras. His father was a clerk in a cloth shop. His mother contributed to the family income by singing at a local temple. Ramanujan studied at the local English language school, displaying his talent and interest for mathematics. At 13 he mastered a textbook used by college students. When he was 15, a university student lent him a copy of *Synopsis of Pure Mathematics*. Ramanujan decided to work out the over 6000 results in this book, stated without proof or explanation, writing on sheets later collected to form notebooks. He graduated from high school in 1904, winning a scholarship to the University of Madras. Enrolling in a fine arts curriculum, he neglected his subjects other than mathematics and lost his scholarship. He failed to pass examinations at the university four times from 1904 to 1907, doing well only in mathematics. During this time he filled his notebooks with original writings, sometimes rediscovering already published work and at other times making new discoveries.

Without a university degree, it was difficult for Ramanujan to find a decent job. To survive, he had to depend on the goodwill of his friends. He tutored students in mathematics, but his unconventional ways of thinking and failure to stick to the syllabus caused problems. He was married in 1909 in an arranged marriage to a young woman 9 years his junior. Needing to support himself and his wife, he moved to Madras and sought a job. He showed his notebooks of mathematical writings to his potential employers, but the books bewildered them. However, a professor at the Presidency College recognized his genius and supported him for a while, and in 1912 he found work as an accounts clerk, earning a small salary.

Ramanujan continued his mathematical work during this time and published his first paper in 1910 in an Indian journal. He realized that his work was beyond that of Indian mathematicians and decided to write to leading English mathematicians. The first mathematicians he wrote to turned down his request for help. But in January 1913 he wrote to G. H. Hardy, who was inclined to turn Ramanujan down, but the mathematical statements in the letter, although stated without proof, puzzled Hardy. He decided to examine them closely with the help of his colleague and collaborator J. E. Littlewood. They decided, after careful study, that Ramanujan was probably a genius, since his statements "could only be written down by a mathematician of the highest class; they must be true, because if they were not true, no one would have the imagination to invent them."

Hardy arranged a scholarship for Ramanujan, bringing him to England in 1914. Hardy personally tutored him in mathematical analysis, and they collaborated for 5 years, proving significant theorems about

3.4

Recursive Algorithms

INTRODUCTION

Sometimes we can reduce the solution to a problem with a particular set of input to the solution of the same problem with smaller input values. For instance, the problem of finding the greatest common divisor of two positive integers a and b where $b > a$ can be reduced to finding the greatest common divisor of a pair of smaller integers, namely, $b \textbf{ mod } a$ and a, since $\gcd(b \textbf{ mod } a, a) = \gcd(a, b)$. When such a reduction can be done, the solution to the original problem can be found with a sequence of reductions, until the problem has been reduced to some initial case for which the solution is known. For instance, for finding the greatest common divisor, the reduction continues until the smaller of the two numbers is zero, since $\gcd(a, 0) = a$ when $a > 0$.

We will see that algorithms that successively reduce a problem to the same problem with smaller input are used to solve a wide variety of problems.

DEFINITION 1. An algorithm is called *recursive* if it solves a problem by reducing it to an instance of the same problem with smaller input.

We will describe several different recursive algorithms in the following examples. The first example shows how a recursive algorithm can be constructed to evaluate a function from its recursive definition.

EXAMPLE 1 Give a recursive algorithm for computing a^n where a is a nonzero real number and n is a nonnegative integer.

Solution: We can base a recursive algorithm on the recursive definition of a^n. This definition states that $a^{n+1} = a \cdot a^n$ for $n > 0$ and the initial condition $a^0 = 1$. To find a^n, successively use the recursive condition to reduce the exponent until it becomes zero. We give this procedure in Algorithm 1. ■

---

the number of partitions of integers. During this time, Ramanujan made important contributions to number theory and also worked on continued fractions, infinite series, and elliptic functions. Ramanujan had amazing insight involving certain types of functions and series, but his purported theorems on prime numbers were often wrong, illustrating his vague idea of what constitutes a correct proof. He was one of the youngest members ever appointed a Fellow of the Royal Society. Unfortunately, in 1917 Ramanujan became extremely ill. At the time, it was thought that he had trouble with the English climate and had contracted tuberculosis. It is now thought that he suffered from a vitamin deficiency, brought on by Ramanujan's strict vegetarianism and shortages in wartime England. He returned to India in 1919, continuing to do mathematics even when confined to his bed. He was religious and thought his mathematical talent came from his family deity, Namagiri. He considered mathematics and religion to be linked. He said that "an equation for me has no meaning unless it expresses a thought of God." His short life came to an end in April 1920, when he was 32 years old. Ramanujan left several notebooks of unpublished results. The writings in these notebooks illustrate Ramanujan's insights but are quite sketchy. Several mathematicians have devoted many years of study to explaining and justifying the results in these notebooks.

ALGORITHM 1 **A Recursive Algorithm for Computing a^n.**

```
procedure power(a: nonzero real number, n: nonnegative integer)
if $n = 0$ then power(a, n) := 1
else power(a, n) := $a * power(a, n - 1)$
```

Next we give a recursive algorithm for finding greatest common divisors.

EXAMPLE 2 Give a recursive algorithm for computing the greatest common divisor of two nonnegative integers a and b with $a < b$.

Solution: We can base a recursive algorithm on the reduction $\gcd(a, b) = \gcd(b \textbf{ mod } a, a)$ and the condition $\gcd(0, b) = b$ when $b > 0$. This produces the procedure in Algorithm 2. ■

ALGORITHM 2 **A Recursive Algorithm for Computing $\gcd(a, b)$.**

```
procedure gcd(a, b: nonnegative integers with $a < b$)
if $a = 0$ then gcd(a, b) := b
else gcd(a, b) := gcd(b mod a, a)
```

We will now give recursive versions of searching algorithms.

EXAMPLE 3 Express the linear search algorithm as a recursive procedure.

Solution: To *search* for x in the search sequence $a_1, a_2, \ldots, a_n$, at the ith step of the algorithm x and a_i are compared. If x equals a_i then i is the location of x. Otherwise, the search for x is reduced to a search in a sequence with one fewer element, namely, the sequence $a_{i+1}, \ldots, a_n$. We can now give a recursive procedure.

Let *search*(i, j, x) be the procedure that searches for x in the sequence $a_i, a_{i+1}, \ldots, a_j$. The input to the procedure consists of the triple $(1, n, x)$. The procedure terminates at a step if the first term of the remaining sequence is x or if there is only one term of the sequence and this is not x. If x is not the first term and there are additional terms, the same procedure is carried out but with a search sequence of one fewer term, obtained by deleting the first term of the search sequence. ■

ALGORITHM 3 **A Recursive Sequential Search Algorithm.**

```
procedure search(i, j, x)
if $a_i = x$ then
    location := i
else if $i = j$ then
    location := 0
else
    search(i + 1, j, x)
```

EXAMPLE 4 Construct a recursive version of a binary search algorithm.

Solution: Suppose we want to locate x in the sequence $a_1, a_2, \ldots, a_n$. To perform a binary search, we begin by comparing x with the middle term, $a_{\lfloor (n+1)/2 \rfloor}$. Our algorithm will terminate if x equals this term. Otherwise, we reduce the search to a smaller search sequence, namely, the first half of the sequence if x is smaller than the middle term of the original sequence, and the second half otherwise. We have reduced the solution of the search problem to the solution of the same problem with a sequence approximately half as long. We express this recursive version of a binary search algorithm as Algorithm 4. ■

ALGORITHM 4 **A Recursive Binary Search Algorithm.**

```
procedure binary search($x, i, j$)
$m := \lfloor (i + j)/2 \rfloor$
if $x = a_m$ then
    location := $m$
else if ($x < a_m$ and $i < m$) then
    binary search($x, i, m - 1$)
else if ($x > a_m$ and $j > m$) then
    binary search($x, m + 1, j$)
else location := 0
```

RECURSION AND ITERATION

A recursive definition expresses the value of a function at a positive integer in terms of the values of the function at smaller integers. This means that we can devise a recursive algorithm to evaluate a recursively defined function at a positive integer.

EXAMPLE 5 The following recursive procedure gives the value of $n!$ when the input is a positive integer n. ■

ALGORITHM 5 **A Recursive Procedure for Factorials.**

```
procedure factorial($n$: positive integer)
if $n = 1$ then
    factorial($n$) := 1
else
    factorial($n$) := $n \cdot$ factorial($n - 1$)
```

There is another way to evaluate the factorial function at an integer from its recursive definition. Instead of successively reducing the computation to the evaluation of the function at smaller integers, we can start with the value of the function at 1 and successively apply the recursive definition to find the values of the function at successive larger integers. Such a procedure is called **iterative.** In other words, to find $n!$ using an iterative procedure, we start with 1, the value of the factorial function at 1, and multiply successively by each positive integer less than or equal to n. This procedure is shown in Algorithm 6.

ALGORITHM 6 An Iterative Procedure for Factorials.

```
procedure iterative factorial($n$: positive integer)
$x := 1$
for $i := 1$ to $n$
    $x := i * x$
{$x$ is $n!$}
```

After this code has been executed, the value of the variable x is $n!$. For instance, going through the loop six times gives $6! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 = 720$.

Often an iterative approach for the evaluation of a recursively defined sequence requires much less computation than a procedure using recursion (unless special-purpose recursive machines are used). This is illustrated by the iterative and recursive procedures for finding the nth Fibonacci number. The recursive procedure is given first.

ALGORITHM 7 A Recursive Algorithm for Fibonacci Numbers.

```
procedure fibonacci($n$: nonnegative integer)
if $n = 0$ then fibonacci(0) := 0
else if $n = 1$ then fibonacci(1) := 1
else fibonacci($n$) := fibonacci($n - 1$) + fibonacci($n - 2$)
```

When we use a recursive procedure to find f_n, we first express f_n as $f_{n-1} + f_{n-2}$. Then we replace both of these Fibonacci numbers by the sum of two previous Fibonacci numbers, and so on. When f_1 or f_0 arises, it is replaced by its value.

Note that at each stage of the recursion, until f_1 or f_0 is obtained, the number of Fibonacci numbers to be evaluated has doubled. For instance, when we find f_4 using this recursive algorithm, we must carry out all the computations illustrated in the tree diagram in Figure 1. This tree consists of a root labeled with f_4, and branches from the root to vertices labeled with the two Fibonacci numbers f_3 and f_2 that occur in the reduction of the computation of f_4. Each subsequent reduction produces two branches in the tree. This branching ends when f_0 and f_1 are reached. The reader can verify that this algorithm requires $f_{n+1} - 1$ additions to find f_n.

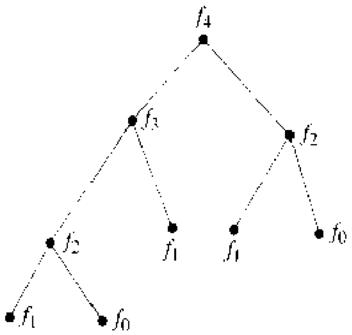


FIGURE 1 Evaluating f_4 Recursively.

Now consider the amount of computation required to find f_n using the following iterative approach.

ALGORITHM 8 An Iterative Algorithm for Computing Fibonacci Numbers.

```
procedure iterative fibonacci(n: nonnegative integer)
if $n = 0$ then $y := 0$
else
begin
    $x := 0$
    $y := 1$
    for $i := 1$ to $n - 1$
    begin
        $z := x + y$
        $x := y$
        $y := z$
    end
end
{$y$ is the $n$th Fibonacci number}
```

This procedure initializes x as $f_0 = 0$ and y as $f_1 = 1$. When the loop is traversed, the sum of x and y is assigned to the auxiliary variable z. Then x is assigned the value of y and y is assigned the value of the auxiliary variable z. Therefore, after going through the loop the first time, it follows that x equals f_1 and y equals $f_0 + f_1 = f_2$. Furthermore, after going through the loop $n - 1$ times, x equals f_{n-1} and y equals f_n (the reader should verify this statement). Only $n - 1$ additions have been used to find f_n with this iterative approach when $n > 1$. Consequently, this algorithm requires far less computation than does the recursive algorithm.

We have shown that a recursive algorithm may require far more computation than an iterative one when a recursively defined function is evaluated. It is sometimes preferable to use a recursive procedure even if it is less efficient than the iterative procedure. In particular, this is true when the recursive approach is easily implemented and the iterative approach is not. (Also, machines designed to handle recursion may be available that eliminate the advantage of using iteration.)

Exercises

1. Give a recursive algorithm for computing nx whenever n is a positive integer and x is an integer.

2. Give a recursive algorithm for finding the sum of the first n positive integers.

3. Give a recursive algorithm for finding the sum of the first n odd positive integers.

4. Give a recursive algorithm for finding the maximum of a finite set of integers.

5. Give a recursive algorithm for finding the minimum of a finite set of integers.

6. Devise a recursive algorithm for finding x^n **mod** m whenever n, x, and m are positive integers.

7. Give a recursive algorithm for finding $n!$ **mod** m whenever n and m are positive integers.

8. Give a recursive algorithm for finding a **mode** of a list of integers. (A **mode** is an element in the list that occurs at least as often as every other element.)

9. Devise a recursive algorithm for computing the greatest common divisor of two nonnegative integers a and b with $a < b$ if $\gcd(a, b) = \gcd(a, b - a)$.

10. Devise a recursive algorithm to find a^{2^n} where a is a real number and n is a positive integer. [*Hint:* Use the equality $a^{2^{n+1}} = (a^{2^n})^2$.]

11. How does the number of multiplications used by the algorithm in Exercise 10 compare to the number of multiplications used by Algorithm 1 to evaluate a^{2^n}?

***12.** Use the algorithm in Exercise 10 to devise an algorithm for evaluating a^n when n is a nonnegative integer. (*Hint:* Use the binary expansion of n.)

***13.** How does the number of multiplications used by the algorithm in Exercise 12 compare to the number of multiplications used by Algorithm 1 to evaluate a^n?

14. How many additions are used by the recursive and iterative algorithms given in Algorithms 7 and 8, respectively, to find the Fibonacci number f_7?

15. Devise a recursive algorithm to find the nth term of the sequence defined by $a_0 = 1$, $a_1 = 2$, and $a_n = a_{n-1} \cdot a_{n-2}$, for $n = 2, 3, 4, \ldots$.

16. Devise an iterative algorithm to find the nth term of the sequence defined in Exercise 15.

17. Is the recursive algorithm or the iterative algorithm for finding the sequence in Exercise 15 more efficient?

18. Devise a recursive algorithm to find the nth term of the sequence defined by $a_0 = 1$, $a_1 = 2$, $a_2 = 3$, and $a_n = a_{n-1} + a_{n-2} + a_{n-3}$, for $n = 3, 4, 5, \ldots$.

19. Devise an iterative algorithm to find the nth term of the sequence defined in Exercise 18.

20. Is the recursive algorithm or the iterative algorithm for finding the sequence in Exercise 18 more efficient?

21. Give iterative and recursive algorithms for finding the nth term of the sequence defined by $a_0 = 1$, $a_1 = 3$, $a_2 = 5$, and $a_n = a_{n-1} \cdot a_{n-2}^2 \cdot a_{n-3}^3$. Which is more efficient?

22. Give a recursive algorithm to find the number of partitions of a positive integer based on the recursive definition given in Exercise 35 in Section 3.3.

23. Give a recursive algorithm for finding the reversal of a bit string. (See the definition of the reversal of a bit string in the preamble of Exercise 26 in Section 3.3.)

24. Give a recursive algorithm for finding the string w^i, the concatenation of i copies of w, when w is a bit string.

25. Give a recursive algorithm for computing values of the Ackermann function. (*Hint:* See the preamble to Exercise 36 in Section 3.3.)

3.5

Program Correctness

INTRODUCTION

Suppose that we have designed an algorithm to solve a problem and have written a program to implement it. How can we be sure that the program always produces the correct answer? After all the bugs have been removed so that the syntax is correct, we can test the program with sample input. It is not correct if an incorrect result is produced for any sample input. But even if the program gives the correct answer for all sample input, it may not always produce the correct answer (unless all possible input has been tested). We need a proof to show that the program *always* gives the correct output.

Program verification, the proof of correctness of programs, uses the rules of inference and proof techniques described in this chapter, including mathematical induction. Since an incorrect program can lead to disastrous effects, a large amount of methodology has been constructed for verifying programs. Efforts have been devoted to automating program verification so that it can be carried out using a computer. However, only limited progress has been made toward this goal. Indeed, some mathematicians and theoretical computer scientists argue that it will never be realistic to mechanize the proof of correctness of complex programs.

Some of the concepts and methods used to prove that programs are correct will be introduced in this section. However, a complete methodology for program verification will not be developed in this book. This section is meant to be a brief introduction to the area of program verification, which ties together the rules of logic, proof techniques, and the concept of an algorithm.

PROGRAM VERIFICATION

A program is said to be **correct** if it produces the correct output for every possible input. A proof that a program is correct consists of two parts. The first part shows that the correct answer is obtained if the program terminates. This part of the proof establishes the **partial correctness** of the program. The second part of the proof shows that the program always terminates.

To specify what it means for a program to produce the correct output, two propositions are used. The first is the **initial assertion,** which gives the properties that the input values must have. The second is the **final assertion,** which gives the properties that the output of the program should have, if the program did what was intended. The appropriate initial and final assertions must be provided when a program is checked.

DEFINITION 1. **A program, or program segment, S is said to be *partially correct with respect to* the initial assertion p and the final assertion q if whenever p is true for the input values of S and S terminates, then q is true for the output values of S. The notation $p\{S\}q$ indicates that the program, or program segment, S is partially correct with respect to the initial assertion p and the final assertion q. *Note:* The notation $p\{S\}q$ is known as a *Hoare triple* after Tony Hoare, who introduced the concept of partial correctness.**

Note that the notion of partial correctness has nothing to do with whether a program terminates; it focuses only on whether the program does what it is expected to do if it terminates.

A simple example illustrates the concepts of initial and final assertions.

EXAMPLE 1 Show that the program segment

$$y := 2$$
$$z := x + y$$

is correct with respect to the initial assertion p: $x = 1$ and the final assertion q: $z = 3$.

Solution: Suppose that p is true, so that $x = 1$ as the program begins. Then y is assigned the value 2, and z is assigned the sum of the values of x and y, which is 3. Hence, S is correct with respect to the initial assertion p and the final assertion q. Thus, $p\{S\}q$ is true. ■

RULES OF INFERENCE

A useful rule of inference proves that a program is correct by splitting the program into a series of subprograms and then showing that each subprogram is correct.

C. Anthony R. Hoare (born 1934). Tony Hoare is currently Professor of Computer Science at Oxford University, England, and is a Fellow of the Royal Society. Hoare has made many important contributions to the theory of programming languages and to programming methodology. He was the first person to define a programming language based on how programs could be proved to be correct with respect to their specifications. Hoare is also the creator of the quick sort, one of the most commonly used and studied sorting algorithms (see the exercise set in Section 8.4). Hoare is a noted writer in the technical and social aspects of computer science.

Suppose that the program S is split into subprograms S_1 and S_2. Write $S = S_1; S_2$ to indicate that S is made up of S_1 followed by S_2. Suppose that the correctness of S_1 with respect to the initial assertion p and final assertion q, and the correctness of S_2 with respect to the initial assertion q and the final assertion r, have been established. It follows that if p is true and S_1 is executed and terminates, then q is true; and if q is true, and S_2 executes and terminates, then r is true. Thus, if p is true and $S = S_1; S_2$ is executed and terminates, then r is true. This rule of inference, called the **composition rule,** can be stated as

$$\begin{array}{l} p\{S_1\}q \\ q\{S_2\}r \\ \hline \therefore p\{S_1; S_2\}r. \end{array}$$

This rule of inference will be used later in this section.

Next, some rules of inference for program segments involving conditional statements and loops will be given. Since programs can be split into segments for proofs of correctness, this will let us verify many different programs.

CONDITIONAL STATEMENTS

First, rules of inference for conditional statements will be given. Suppose that a program segment has the form

```
if condition then
    S
```

where S is a block of statements. Then S is executed if *condition* is true, and it is not executed when *condition* is false. To verify that this segment is correct with respect to the initial assertion p and final assertion q, two things must be done. First, it must be shown that when p is true and *condition* is also true, then q is true after S terminates. Second, it must be shown that when p is true and *condition* is false, then q is true (since in this case S does not execute).

This leads to the following rule of inference:

$$\begin{array}{l} (p \wedge \mathit{condition})\{S\}q \\ (p \wedge \neg \mathit{condition}) \rightarrow q \\ \hline \therefore p\{\textbf{if } \mathit{condition} \textbf{ then } S\}q. \end{array}$$

The following example illustrates how this rule of inference is used.

EXAMPLE 2 Verify that the program segment

```
if x > y then
    y := x
```

is correct with respect to the initial assertion **T** and the final assertion $y \geq x$.

Solution: When the initial assertion is true and $x > y$, the assignment $y := x$ is carried out. Hence, the final assertion, which asserts that $y \geq x$, is true in this case. Moreover, when the initial assertion is true and $x > y$ is false, so that $x \leq y$, the final assertion is again true. Hence, using the rule of inference for program segments of this type, this program is correct with respect to the given initial and final assertions. ■

Similarly, suppose that a program has a statement of the form

```
**if** *condition* **then**
    $S_1$
**else**
    $S_2$
```

If *condition* is true, then S_1 executes; if *condition* is false, then S_2 executes. To verify that this program segment is correct with respect to the initial assertion p and the final assertion q, two things must be done. First, it must be shown that when p is true and *condition* is true, then q is true after S_1 terminates. Second, it must be shown that when p is true and *condition* is false, then q is true after S_2 terminates. This leads to the following rule of inference:

$$\begin{array}{l} (p \wedge \mathit{condition})\{S_1\}q \\ (p \wedge \neg \mathit{condition})\{S_2\}q \\ \hline \therefore p\{\textbf{if } \mathit{condition} \textbf{ then } S_1 \textbf{ else } S_2\}q. \end{array}$$

The following example illustrates how this rule of inference is used.

EXAMPLE 3 Verify that the program segment

```
**if** $x < 0$ **then**
    $abs := -x$
**else**
    $abs := x$
```

is correct with respect to the initial assertion **T** and the final assertion $abs = |x|$.

Solution: Two things must be demonstrated. First, it must be shown that if the initial assertion is true and $x < 0$, then $abs = |x|$. This is correct, since when $x < 0$ the assignment statement $abs := -x$ sets $abs = -x$, which is $|x|$ by definition when $x < 0$. Second, it must be shown that if the initial assertion is true and $x < 0$ is false, so that $x \geq 0$, then $abs = |x|$. This is also correct, since in this case the program uses the assignment statement $abs := x$, and x is $|x|$ by definition when $x \geq 0$, so that $abs := x$. Hence, using the rule of inference for program segments of this type, this segment is correct with respect to the given initial and final assertions. ■

LOOP INVARIANTS

Next, proofs of correctness of **while** loops will be described. To develop a rule of inference for program segments of the type

```
while condition
    S
```

note that S is repeatedly executed until *condition* becomes false. An assertion that remains true each time S is executed must be chosen. Such an assertion is called a **loop invariant.** In other words, p is a loop invariant if $(p \wedge condition)\{S\}p$ is true.

Suppose that p is a loop invariant. It follows that if p is true before the program segment is executed, p and $\neg condition$ are true after termination, if it occurs. This rule of inference is

$$\begin{array}{l} (p \wedge condition)\{S\}p \\ \hline \therefore p\{\mathbf{while}\ condition\ S\}(\neg condition \wedge p). \end{array}$$

The use of a loop invariant is illustrated in the following example.

EXAMPLE 4 A loop invariant is needed to verify that the program segment

```
i := 1
factorial := 1
while i < n
begin
    i := i + 1
    factorial := factorial * i
end
```

terminates with $factorial = n!$ when n is a positive integer. Let p be the proposition "$factorial := i!$ and $i \leq n$." We will prove that p is a loop invariant using mathematical induction. First, note that p is true before the loop is entered, since at that point $factorial = 1 = 1!$ and $1 \leq n$. Now assume that p is true and $i < n$ after an execution of the loop. Assume that the **while** loop is executed again. First, i is incremented by 1. Thus, i is still less than or equal to n, since by the inductive hypothesis $i < n$ before the loop was entered, and i and n are positive integers. Furthermore, *factorial*, which was $(i-1)!$ by the inductive hypothesis, is set equal to $(i-1)! \cdot i = i!$. Hence, p remains true. Therefore p is a loop invariant. In other words, the assertion $[p \wedge (i < n)]\{S\}p$ is true. It follows that the assertion $p\{\mathbf{while}\ i < n\ S\}[(i \geq n) \wedge p]$ is also true.

Furthermore, the loop terminates after $n - 1$ traversals with $i = n$, since i is assigned the value 1 at the beginning of the program, 1 is added to i at each traversal, and the loop terminates when $i \geq n$. Consequently, at termination $factorial = n!$. ■

A final example will be given to show how the various rules of inference can be used to verify the correctness of a longer program.

EXAMPLE 5 We will outline how to verify the correctness of the program S for computing the product of two integers.

```
procedure multiply(m, n: integers)
     ⎧ if $n < 0$ then $a := -n$
S1   ⎩ else $a := n$
     ⎧ $k := 0$
S2   ⎩ $x := 0$
     ⎧ while $k < a$
     ⎪ begin
S3   ⎨    $x := x + m$
     ⎪    $k := k + 1$
     ⎩ end
     ⎧ if $n < 0$ then product $:= -x$
S4   ⎩ else product $:= x$
```

The goal is to prove that after S is executed, *product* has the value mn. The proof of correctness can be carried out by splitting S into four segments, with $S = S_1; S_2; S_3; S_4$, as shown in the listing of S. The rule of composition can be used to build the correctness proof. Here is how the argument proceeds. The details will be left as an exercise for the reader.

Let p be the initial assertion that m and n are integers. Then, it can be shown that $p\{S_1\}q$ is true, when q is the proposition $p \wedge (a = |n|)$. Next, let r be the proposition $q \wedge (k = 0) \wedge (x = 0)$. It is easily verified that $q\{S_2\}r$ is true. It can be shown that "$x = mk$ and $k \leq a$" is an invariant for the loop in S_3. Furthermore, it is easy to see that the loop terminates after a iterations, with $k = a$, so that $x = ma$ at this point. Since r implies that $x = m \cdot 0$ and $0 \leq a$, the loop invariant is true before the loop is entered. Since the loop terminates with $k = a$, it follows that $r\{S_3\}s$ is true where s is the proposition "$x = ma$ and $a = |n|$." Finally, it can be shown that S_4 is correct with respect to the initial assertion s and final assertion t, where t is the proposition "$product = mn$."

Putting all this together, since $p\{S_1\}q$, $q\{S_2\}r$, $r\{S_3\}s$, and $s\{S_4\}t$ are all true, it follows from the rule of composition that $p\{S\}t$ is true. Furthermore, since all four segments terminate, S does terminate. This verifies the correctness of the program. ■

Exercises

1. Prove that the program segment

   $y := 1$
   $z := x + y$

   is correct with respect to the initial assertion $x = 0$ and the final assertion $z = 1$.

2. Verify that the program segment

   if $x < 0$ **then** $x := 0$

   is correct with respect to the initial assertion **T** and the final assertion $x \geq 0$.

3. Verify that the program segment

```
$x$ := 2
$z$ := $x + y$
**if** $y > 0$ **then**
    $z$ := $z + 1$
**else**
    $z$ := 0
```

is correct with respect to the initial assertion $y = 3$ and the final assertion $z = 6$.

4. Verify that the program segment

```
**if** $x < y$ **then**
    $min$ := $x$
**else**
    $min$ := $y$
```

is correct with respect to the initial assertion **T** and the final assertion $(x \leq y \wedge min = x) \vee (x > y \wedge min = y)$.

*5. Devise a rule of inference for verification of partial correctness of statements of the form

```
**if** *condition* 1 **then**
    $S_1$
**else if** *condition* 2 **then**
    $S_2$
      ⋮
**else**
    $S_n$
```

where $S_1, S_2, \ldots, S_n$ are blocks.

6. Use the rule of inference developed in Exercise 5 to verify that the program

```
**if** $x < 0$ **then**
    $y$ := $-2|x|/x$
**else if** $x > 0$ **then**
    $y$ := $2|x|/x$
**else if** $x = 0$ **then**
    $y$ := 2
```

is correct with respect to the initial assertion **T** and the final assertion $y = 2$.

7. Use a loop invariant to prove that the following program segment for computing the nth power, where n is a positive integer, of a real number x is correct.

```
$power$ := 1
$i$ := 1
**while** $i \leq n$
**begin**
    $power$ := $power * x$
    $i$ := $i + 1$
**end**
```

*8. Prove that the iterative program for finding f_n given in Section 3.4 is correct.

9. Provide all the details in the proof of correctness given in Example 5.

10. Suppose that both the implication $p_0 \rightarrow p_1$ and the program assertion $p_1\{S\}q$ are true. Show that $p_0\{S\}q$ also must be true.

11. Suppose that both the program assertion $p\{S\}q_0$ and the implication $q_0 \rightarrow q_1$ are true. Show that $p\{S\}q_1$ also must be true.

12. The following program computes quotients and remainders.

```
$r$ := $a$
$q$ := 0
**while** $r \geq d$
**begin**
    $r$ := $r - d$
    $q$ := $q + 1$
**end**
```

Verify that it is partially correct with respect to the initial assertion "a and d are positive integers" and the final assertion "q and r are integers such that $a = dq + r$ and $0 \leq r < d$."

13. Use a loop invariant to verify that the Euclidean algorithm (Algorithm 1 in Section 2.4) is partially correct with respect to the initial assertion "a and b are positive integers" and the final assertion "$x = \gcd(a, b)$."

Key Terms and Results

TERMS

theorem: a mathematical assertion that can be shown to be true

conjecture: a mathematical assertion whose truth value is unknown

proof: a demonstration that a theorem is true

lemma: a simple theorem used to prove other theorems

corollary: a proposition that can be proved as a consequence of a theorem that has just been proved

rule of inference: an implication that is a tautology which is then used to draw conclusions from known assertions

fallacy: an implication that is a contingency that is often incorrectly used to draw conclusions

circular reasoning or **begging the question:** reasoning where one or more steps are based on the truth of the statement being proved

vacuous proof: a proof that the implication $p \rightarrow q$ is true based on the fact that p is false

trivial proof: a proof that the implication $p \rightarrow q$ is true based on the fact that q is true

direct proof: a proof that the implication $p \rightarrow q$ is true that proceeds by showing that q must be true when p is true

indirect proof: a proof that the implication $p \rightarrow q$ is true that proceeds by showing that p must be false when q is false

proof by contradiction: a proof that a proposition p is true based on the truth of the implication $\neg p \rightarrow q$ where q is a contradiction

proof by cases: a proof of an implication where the hypothesis is a disjunction of propositions that shows that each hypothesis separately implies the conclusion

counterexample: an element x such that $P(x)$ is false

mathematical induction: a proof technique for statements of the form $\forall n \in \mathbf{N}\, P(n)$ consisting of a basis step and an inductive step

basis step: the proof of $P(1)$ in a proof of $\forall n \in \mathbf{N}\, P(n)$ by mathematical induction

inductive step: the proof of $P(n) \rightarrow P(n+1)$ in a proof of $\forall n \in \mathbf{N}\, P(n)$ by mathematical induction

recursive definition of a function: a definition of a function that specifies an initial set of values and a rule for obtaining values of this function at integers from its values at smaller integers

recursive definition of a set: a definition of a set that specifies an initial set of elements in the set and a rule for obtaining other elements from those in the set

recursive algorithm: an algorithm that proceeds by reducing a problem to the same problem with smaller input

iteration: a procedure based on the repeated use of operations in a loop

program correctness: verification that a procedure always produces the correct result

loop invariant: a property that remains true during every traversal of a loop

initial assertion: the statement specifying the properties of the input values of a program

final assertion: the statement specifying the properties the output values should have if the program worked correctly

RESULTS

The well-ordering property: Every nonempty set of nonnegative integers has a least element.

Principle of mathematical induction: The statement $\forall n\, P(n)$ is true if $P(1)$ is true and $\forall n[P(n) \rightarrow P(n+1)]$ is true.

Second principle of mathematical induction: The statement $\forall n\, P(n)$ is true if $P(1)$ is true and $\forall n[(P(1) \wedge \cdots \wedge P(n)) \rightarrow P(n+1)]$ is true.

Review Questions

1. a) Describe what is meant by a direct proof, an indirect proof, and a proof by contradiction of an implication $p \rightarrow q$.
   b) Give a direct proof, an indirect proof, and a proof by contradiction of the statement: "If n is even, then $n + 4$ is even."
2. a) Describe one way to prove the biconditional $p \leftrightarrow q$.
   b) Prove the statement: "The integer $3n + 2$ is odd if and only if the integer $9n + 5$ is even, where n is an integer."
3. To prove that the statements p_1, p_2, p_3, and p_4 are equivalent, is it sufficient to show that the implications $p_4 \rightarrow p_2$, $p_3 \rightarrow p_1$, and $p_1 \rightarrow p_2$ are valid? If not, provide another set of implications that can be used to show that the four statements are equivalent.
4. a) Suppose that a statement of the form $\forall x\, P(x)$ is false. How can this be proved?
   b) Show that the statement "For every positive integer n, the number $n^2 + 1$ is prime" is false.
5. a) What is the difference between a constructive existence proof and a nonconstructive existence proof?
   b) Show that for every integer n there is an integer greater than n that is not divisible by 3 or 5. Is your existence proof constructive or nonconstructive?
6. a) State the well-ordering property for the set of positive integers.
   b) Use this property to show that every positive integer can be written as the product of primes.
7. a) Can you use the principle of mathematical induction to find a formula for the sum of the first n terms of a sequence?
   b) Can you use the principle of mathematical induction to determine whether a given formula for the sum of the first n terms of a sequence is correct?
   c) Find a formula for the sum of the first n even positive integers, and prove it using mathematical induction.
8. a) For which positive integers n is it true that $11n + 17 \leq 2^n$?
   b) Prove the conjecture you made in part (a) using mathematical induction.

9. **a)** Which amounts of postage can be formed using only 5-cent and 9-cent stamps?
  b) Prove the conjecture you made using mathematical induction.
  c) Prove the conjecture you made using the second principle of mathematical induction.
  d) Find a proof of your conjecture different from the ones you gave in (b) and (c).

10. Give three different examples of proofs that use the second principle of mathematical induction.

11. **a)** Explain why a function is well-defined if it is defined recursively by specifying $f(1)$ and a rule for finding $f(n)$ from $f(n-1)$.
  b) Provide a recursive definition of the function $f(n) = (n+1)!$.

12. **a)** Give a recursive definition of the Fibonacci numbers.
  b) Show that $f_n > \alpha^{n-2}$ whenever $n \geq 3$ where f_n is the nth term of the Fibonacci sequence and $\alpha = (1+\sqrt{5})/2$.

13. **a)** Explain why a sequence a_n is well-defined if it is defined recursively by specifying a_1 and a_2 and a rule for finding a_n from $a_1, a_2, \ldots, a_{n-1}$ for $n = 3, 4, 5, \ldots$.
  b) Find the value of a_n if $a_1 = 1$, $a_2 = 2$, and $a_n = a_{n-1} + a_{n-2} + \cdots + a_1$, for $n = 3, 4, 5, \ldots$.

14. Give two examples of how well-formed formulae are defined recursively for different sets of elements and operators.

15. **a)** Give a recursive definition of the length of a string.
  b) Use the recursive definition from part (a) to prove that $l(xy) = l(x) + l(y)$.

16. **a)** What is a recursive algorithm?
  b) Describe a recursive algorithm for computing the sum of n numbers in a sequence.

17. Describe a recursive algorithm for computing the greatest common divisor of two positive integers.

18. **a)** Does testing a computer program to see whether it produces the correct output for certain input values verify that the program always produces the correct output?
  b) Does showing that a computer program is partially correct with respect to an initial assertion and a final assertion verify that the program always produces the correct output? If not, what else is needed?

19. What techniques can you use to show that a long computer program is partially correct with respect to an initial assertion and a final assertion?

20. What is a loop invariant? How is a loop invariant used?

Supplementary Exercises

1. Prove that the product of two odd numbers is odd.

2. Prove that $\sqrt{5}$ is irrational.

3. Prove or disprove that the sum of two irrational numbers is irrational.

4. Prove or disprove that $n^2 + n + 1$ is prime whenever n is a positive integer.

5. Determine whether the following is a valid argument. If n is greater than 5, then n^2 is greater than 25. Therefore, if n is an integer with n^2 greater than 25, it follows that n is greater than 5.

6. Prove that $n^4 - 1$ is divisible by 5 when n is not divisible by 5. Use a proof by cases, with four different cases—one for each of the nonzero remainders that an integer not divisible by 5 can have when you divide it by 5.

7. Prove that $|xy| = |x||y|$ by cases.

***8.** We define the **Ulam numbers** by setting $u_1 = 1$ and $u_2 = 2$. Furthermore, after determining whether the integers less than n are Ulam numbers, we set n equal to the next Ulam number if it can be written uniquely as the sum of two different Ulam numbers. Note that $u_3 = 3$, $u_4 = 4$, $u_5 = 6$, and $u_6 = 8$.
  a) Find the first 20 Ulam numbers.
  b) Prove that there are infinitely many Ulam numbers.

9. Give a constructive proof that there is a polynomial $P(x)$ such that $P(x_1) = y_1, P(x_2) = y_2, \ldots, P(x_n) = y_n$, where $x_1, \ldots, x_n, y_1, \ldots, y_n$ are real numbers. [*Hint:* Let

$$P(x) = \sum_{i=1}^{n} \left(\prod_{i \neq j} \frac{x - x_j}{x_i - x_j} \right) y_i.]$$

10. Show that $1^3 + 3^3 + 5^3 + \cdots + (2n+1)^3 = (n+1)^2(2n^2 + 4n + 1)$ whenever n is a positive integer.

11. Show that $1 \cdot 2^0 + 2 \cdot 2^1 + 3 \cdot 2^2 + \cdots + n \cdot 2^{n-1} = (n-1) \cdot 2^n + 1$ whenever n is a positive integer.

12. Show that

$$\frac{1}{1 \cdot 3} + \frac{1}{3 \cdot 5} + \cdots + \frac{1}{(2n-1)(2n+1)} = \frac{n}{2n+1}$$

whenever n is a positive integer.

13. Show that

$$\frac{1}{1 \cdot 4} + \frac{1}{4 \cdot 7} + \cdots + \frac{1}{(3n-2)(3n+1)} = \frac{n}{3n+1}$$

whenever n is a positive integer.

14. Use mathematical induction to show that $2^n > n^2 + n$ whenever n is an integer greater than 4.

15. Use mathematical induction to show that $2^n > n^3$ whenever n is an integer greater than 9.

16. Find an integer N such that $2^n > n^4$ whenever n is greater than N. Prove that your result is correct using mathematical induction.

17. Use mathematical induction to prove that $a - b$ is a factor of $a^n - b^n$ whenever n is a positive integer.

18. Use mathematical induction to prove that 9 divides $n^3 + (n+1)^3 + (n+2)^3$ whenever n is a nonnegative integer.

19. An **arithmetic progression** is a sequence of the form $a, a+d, a+2d, \ldots, a+nd$ where a and d are real numbers. Use mathematical induction to prove that the sum of these terms of an arithmetic progression is given by

$$a + (a+d) + \cdots + (a+nd) = (n+1)(2a+nd)/2.$$

20. Suppose that $a_j \equiv b_j \pmod{m}$ for $j = 1, 2, \ldots, n$. Use mathematical induction to prove that

a) $\sum_{j=1}^{n} a_j \equiv \sum_{j=1}^{n} b_j \pmod{m}$

b) $\prod_{j=1}^{n} a_j \equiv \prod_{j=1}^{n} b_j \pmod{m}$

***21.** Determine which Fibonacci numbers are even, and use a form of mathematical induction to prove your conjecture.

***22.** Determine which Fibonacci numbers are divisible by 3. Use a form of mathematical induction to prove your conjecture.

***23.** Prove that $f_k f_n + f_{k+1} f_{n+1} = f_{n+k+1}$ for all nonnegative integers n, where k is a nonnegative integer and f_i denotes the ith Fibonacci number.

The sequence of **Lucas numbers** is defined by $l_0 = 2$, $l_1 = 1$, and $l_n = l_{n-1} + l_{n-2}$ for $n = 2, 3, 4, \ldots$.

24. Show that $f_n + f_{n+2} = l_{n+1}$ whenever n is a positive integer, where f_i and l_i are the ith Fibonacci number and ith Lucas number, respectively.

25. Show that $l_0^2 + l_1^2 + \cdots + l_n^2 = l_n l_{n+1} + 2$ whenever n is a nonnegative integer and l_i is the ith Lucas number.

***26.** Use mathematical induction to show that the product of any n consecutive positive integers is divisible by $n!$. [*Hint:* Use the identity $m(m+1)\cdots(m+n-1)/n! = (m-1)m(m+1)\cdots(m+n-2)/n! + m(m+1)\cdots(m+n-2)/(n-1)!$.]

27. Use mathematical induction to show that $(\cos x + i \sin x)^n = \cos nx + i \sin nx$ whenever n is a positive integer. [*Hint:* Use the identities $\cos(a+b) = \cos a \cos b - \sin a \sin b$ and $\sin(a+b) = \sin a \cos b + \cos a \sin b$.]

***28.** Use mathematical induction to show that $\sum_{j=1}^{n} \cos jx = \cos[(n+1)x/2] \sin(nx/2) \sin(x/2)$ whenever n is a positive integer and $\sin(x/2) \neq 0$.

The **McCarthy 91** function is defined using the rule

$$M(n) = \begin{cases} n - 10 & \text{if } n > 100 \\ M(M(n+11)) & \text{if } n \leq 100 \end{cases}$$

for all positive integers n.

29. By successively using the defining rule for $M(n)$, find

a) $M(102)$. **b)** $M(101)$. **c)** $M(99)$.
d) $M(97)$. **e)** $M(87)$. **f)** $M(76)$.

****30.** Show that the function $M(n)$ is a well-defined function from the set of positive integers to the set of positive integers. [*Hint:* Prove that $M(n) = 91$ for all positive integers n with $n \leq 101$.]

31. Is the following proof that

$$\frac{1}{1 \cdot 2} + \frac{1}{2 \cdot 3} + \cdots + \frac{1}{(n-1)n} = \frac{3}{2} - \frac{1}{n},$$

whenever n is a positive integer, correct? Justify your answer.

BASIS STEP: The result is true when $n = 1$ since

$$\frac{1}{1 \cdot 2} = \frac{3}{2} - \frac{1}{1}.$$

INDUCTIVE STEP: Assume that the result is true for n. Then

$$\begin{aligned} \frac{1}{1 \cdot 2} + \frac{1}{2 \cdot 3} + \cdots + \frac{1}{(n-1)n} + \frac{1}{n(n+1)} &= \frac{3}{2} - \frac{1}{n} + \left(\frac{1}{n} - \frac{1}{n+1}\right) \\ &= \frac{3}{2} - \frac{1}{n+1}. \end{aligned}$$

Hence, the result is true for $n + 1$ if it is true for n. This completes the proof.

***32.** A jigsaw puzzle is put together by successively joining pieces that fit together into blocks. A move is made each time a piece is added to a block, or when two blocks are joined. Use the second form of mathematical induction to prove that no matter how the moves are carried out, exactly $n - 1$ moves are required to assemble a puzzle with n pieces.

***33.** Show that n circles divide the plane into $n^2 - n + 2$ regions if every two circles intersect in exactly two points and no three circles contain a common point.

***34.** Show that n planes divide three-dimensional space into $(n^3 + 5n + 6)/6$ regions if any three of these planes have a point in common and no four contain a common point.

***35.** Use the well-ordering property to show that $\sqrt{2}$ is irrational. (*Hint:* Assume that $\sqrt{2}$ is rational. Show that the set of positive integers of the form $b\sqrt{2}$ has a least element a. Then show that $a\sqrt{2} - a$ is a smaller positive integer of this form.)

36. A set is **well-ordered** if every nonempty subset of this set has a least element. Determine whether each of the following sets is well-ordered.

a) the set of integers
b) the set of integers greater than -100
c) the set of positive rationals
d) the set of positive rationals with denominator less than 100

***37.** Show that the well-ordering property can be proved when the principle of mathematical induction is taken as an axiom.

***38.** Show that the first and second principles of mathematical induction are equivalent; that is, each can be shown to be valid from the other.

39. a) Show that if $a_1, a_2, \ldots, a_n$ are positive integers, then $\gcd(a_1, a_2, \ldots, a_{n-1}, a_n) = \gcd(a_1, a_2, \ldots, a_{n-2}, \gcd(a_{n-1}, a_n))$.
b) Use part (a), together with the Euclidean algorithm, to develop a recursive algorithm for computing the greatest common divisor of a set of n positive integers.

***40.** Describe a recursive algorithm for writing the greatest common divisor of n positive integers as a linear combination of these integers.

41. Find an explicit formula for $f(n)$ if $f(1) = 1$ and $f(n) = f(n-1) + 2n - 1$ for $n \geq 2$. Prove your result using mathematical induction.

****42.** Give a recursive definition of the set of bit strings that contain twice as many 0s as 1s.

43. Let S be the set of bit strings defined recursively by $\lambda \in S$ and $0x \in S$, $x1 \in S$ if $x \in S$, where λ is the empty string.
a) Find all strings in S of length not exceeding five.
b) Give an explicit description of the elements of S.

44. Let S be the set of strings defined recursively by $abc \in S$, $bac \in S$, $acb \in S$, and $abcx \in S$; $abxc \in S$, $axbc \in S$, $xabc \in S$ if $x \in S$.
a) Find all elements of S of length eight or less.
b) Show that every element of S has a length divisible by three.

The set B of all **balanced strings of parentheses** is defined recursively by $\lambda \in B$, where λ is the empty string; $(x) \in B$, $xy \in B$ if $x, y \in B$.

45. Find all balanced strings of parentheses with four or fewer symbols.

46. Use induction to show that if x is a balanced string of parentheses, then the number of left parentheses equals the number of right parentheses in x.

Define the function N on the set of strings of parentheses by

$$N(\lambda) = 0, N(() = 1, N()) = -1,$$
$$N(uv) = N(u) + N(v),$$

where λ is the empty string, and u and v are strings. It can be shown that N is well-defined.

47. Find
a) $N(())$ b) $N()))())(()$
c) $N((()(())$ d) $N(()((()))(()))$

****48.** Show that a string w of parentheses is balanced if and only if $N(w) = 0$ and $N(u) \geq 0$ whenever u is a prefix of w, that is, $w = uv$.

***49.** Give a recursive algorithm for finding all balanced strings of parentheses containing n or fewer symbols.

50. Give a recursive algorithm for finding the greatest common divisor of two nonnegative integers a and b with $a \leq b$, based on the fact that $\gcd(a, b) = a$ if $a = b$, $\gcd(a, b) = 2\gcd(a/2, b/2)$ if a and b are even, $\gcd(a, b) = (a/2, b)$ if a is even and b is odd, and $\gcd(a, b) = \gcd(b - a, a)$ if a and b are odd.

51. Verify the program segment

```
if $x > y$ then
    $x := y$
```

with respect to the initial assertion **T** and the final assertion $x \leq y$.

***52.** Develop a rule of inference for verifying recursive programs and use it to verify the recursive program for computing factorials given in Section 3.4.

Computer Projects

WRITE PROGRAMS WITH THE FOLLOWING INPUT AND OUTPUT.

1. Given a geometric progression $a, ar, ar^2, \ldots, ar^n$, find the sum of its terms.

2. Given a nonnegative integer n, find the sum of the n smallest positive integers.

****3.** Given a $2^n \times 2^n$ chessboard with one square missing, construct a tiling of this chessboard using L-shaped pieces.

****4.** Generate all well-formed formulae for expressions involving the variables x, y, and z and the operators $\{+, *, /, -\}$ with n or fewer symbols.

****5.** Generate all well-formed formulae for propositions with n or fewer symbols where each symbol is **T**, **F**, one of the propositional variables p and q, or an operator from $\{\neg, \vee, \wedge, \rightarrow, \leftrightarrow\}$.

6. Given a string, find its reversal.

7. Given a real number a and a nonnegative integer n, find a^n using recursion.

8. Given a real number a and a nonnegative integer n, find a^{2^n} using recursion.

***9.** Given a real number a and a nonnegative integer n, find a^n using the binary expansion of n and a recursive algorithm for computing a^{2^k}.

10. Given two integers not both zero, find their greatest common divisor using recursion.

11. Given a list of integers and an element x, locate x in this list using a recursive implementation of a linear search.

12. Given a list of integers and an element x, locate x in this list using a recursive implementation of a binary search.

13. Given a nonnegative integer n, find the nth Fibonacci number using iteration.

14. Given a nonnegative integer n, find the nth Fibonacci number using recursion.

15. Given a positive integer, find the number of partitions of this integer. (See Exercise 35 of Section 3.3.)

16. Given positive integers m and n, find $A(m, n)$, the value of Ackermann's function at the pair (m, n). (See the preamble to Exercise 36 of Section 3.3.)

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. Verify Goldbach's conjecture, which states that every even positive integer n is the sum of two primes, for $n \leq 10{,}000$.

2. Find the smallest prime factor of $n! + 1$ for all positive integers n with $n \leq 20$.

3. Find the smallest set of n consecutive composite integers for each positive integer n with $n \leq 10$.

4. An old unsettled conjecture states that there are infinitely many *twin primes*, that is, primes that differ by two. How many twin primes can you find?

5. Determine which Fibonacci numbers are divisible by 5, which are divisible by 7, and which are divisible by 11. Prove that your conjectures are correct.

6. Construct tilings using L-shaped pieces of various 16×16, 32×32, and 64×64 chessboards with one square missing.

7. Explore which $m \times n$ chessboards can be completely covered by L-shaped pieces. Can you make a conjecture that answers this question?

8. The notorious $3x + 1$ conjecture (also known as the *Collatz conjecture* and by many other names) states that no matter which integer x you start with, iterating the function $f(x)$, where $f(x) = x/2$ if x is even and $f(x) = 3x + 1$ if x is odd, always produces the integer 1. Verify this conjecture for as many positive integers as possible.

9. Which values of Ackermann's function are small enough that you are able to compute them?

10. Compare either the number of operations or the time needed to compute Fibonacci numbers recursively versus that needed to compute them iteratively.

Writing Projects

RESPOND TO THE FOLLOWING WITH ESSAYS USING OUTSIDE SOURCES.

1. Describe the origins of mathematical induction. Who were the first people to use it and to which problems did they apply it?

2. In the past 20 years, several important theorems have been proved based on extensive computer computations. Discuss the validity of such proofs and describe the controversy surrounding proofs based on computer calculations.

3. *Logic programming* operates on statements expressed using quantifiers, predicates, and logical connectives using rules of inference. Explain the fundamental concepts of logic programming and how it is used in artificial intelligence. Illustrate its use with the programming language PROLOG.

4. "Automated theorem proving" is the task of using computers to mechanically proof theorems. Discuss the goals and applications of automated theorem proving and the progress made in developing automated theorem provers.

5. Describe the basic rules of WFF'N PROOF, The Game of Modern Logic, developed by Layman Allen. (Because this was published in the mid-1960s, you may have to have a scavenger hunt to find a set.) Give examples of some of the games included in WFF'N PROOF.

6. The L-shaped pieces used in the exercises of Section 3.2 are examples of *polyominos*, introduced by Golomb in 1954. Describe some of the problems and associated results concerning tiling chessboards with polyominos.

7. Discuss the uses of Ackermann's function both in the theory of recursive definitions and in the analysis of the complexity of algorithms for set unions.
8. Describe some of the logical problems found in the writings of Lewis Carroll and show how rules of inference are used to solve these problems.
9. Discuss some of the various methodologies used to establish the correctness of programs and compare them to Hoare's methods described in Section 3.5.
10. Explain how the ideas and concepts of program correctness can be extended to prove that operating systems are secure.

4

Counting

Combinatorics, the study of arrangements of objects, is an important part of discrete mathematics. This subject was studied as long ago as the seventeenth century, when combinatorial questions arose in the study of gambling games. Enumeration, the counting of objects with certain properties, is an important part of combinatorics. We must count objects to solve many different types of problems. For instance, counting is used to determine the complexity of algorithms. Counting is also required to determine whether there are enough telephone numbers or Internet protocol addresses to meet demand. Furthermore, counting techniques are used extensively when probabilities of events are computed.

The basic rules of counting, which we will study in Section 4.1, can solve a tremendous variety of problems. For instance, we can use these rules to enumerate the different phone numbers possible in the United States, the allowable passwords on a computer system, and the different orders in which the runners in a race can finish. Another important combinatorial tool is the pigeonhole principle, which we will study in Section 4.2. This states that when objects are placed in boxes and there are more objects than boxes, then there is a box containing at least two objects. For instance, we can use this principle to show that among a set of 15 or more students, at least 3 were born on the same day of the week.

We can phrase many counting problems in terms of ordered or unordered arrangements of the objects of a set. These arrangements, called permutations and combinations, are used in many counting problems. For instance, suppose the 100 top finishers on a competitive exam taken by 2000 students are invited to a banquet. We can enumerate the possible sets of 100 students that will be invited, as well as the ways the top 10 prizes can be awarded.

We can analyze gambling games, such as poker, using counting techniques. We can also use these techniques to determine the probabilities of winning lotteries, such as the probability a person will win a lottery where 6 numbers are chosen from the first 48 positive integers.

Another problem in combinatorics involves generating all the arrangements of a specified kind. This is often important in computer simulations. We will devise algorithms to generate arrangements of various types.

4.1

The Basics of Counting

INTRODUCTION

A password on a computer system consists of six, seven, or eight characters. Each of these characters must be a digit or a letter of the alphabet. Each password must contain at least one digit. How many such passwords are there? The techniques needed to answer this question and a wide variety of other counting problems will be introduced in this section.

Counting problems arise throughout mathematics and computer science. For example, we must count the successful outcomes of experiments and all the possible outcomes of these experiments to determine probabilities of discrete events. We need to count the number of operations used by an algorithm to study its time complexity.

We will introduce the basic techniques of counting in this section. These methods serve as the foundation for almost all counting techniques.

BASIC COUNTING PRINCIPLES

We will present two basic counting principles. Then we will show how they can be used to solve many different counting problems.

THE SUM RULE If a first task can be done in n_1 ways and a second task in n_2 ways, and if these tasks cannot be done at the same time, then there are $n_1 + n_2$ ways to do either task.

The following example illustrates how the sum rule is used.

EXAMPLE 1 Suppose that either a member of the mathematics faculty or a student who is a mathematics major is chosen as a representative to a university committee. How many different choices are there for this representative if there are 37 members of the mathematics faculty and 83 mathematics majors?

Solution: The first task, choosing a member of the mathematics faculty, can be done in 37 ways. The second task, choosing a mathematics major, can be done in 83 ways. From the sum rule it follows that there are $37 + 83 = 120$ possible ways to pick this representative. ■

We can extend the sum rule to more than two tasks. Suppose that the tasks $T_1, T_2, \ldots, T_m$ can be done in $n_1, n_2, \ldots, n_m$ ways, respectively, and no two of these tasks can be done at the same time. Then the number of ways to do one of these tasks is $n_1 + n_2 + \cdots + n_m$. This extended version of the sum rule is often useful in counting problems, as Examples 2 and 3 show. This version of the sum rule can be proved using mathematical induction from the sum rule for two sets. (This is Exercise 53 at the end of the section.)

EXAMPLE 2 A student can choose a computer project from one of three lists. The three lists contain 23, 15, and 19 possible projects, respectively. How many possible projects are there to choose from?

Solution: The student can choose a project from the first list in 23 ways, from the second list in 15 ways, and from the third list in 19 ways. Hence, there are $23 + 15 + 19 = 57$ projects to choose from. ■

EXAMPLE 3 What is the value of k after the following code has been executed?

```
$k := 0$
for $i_1 := 1$ to $n_1$
    $k := k + 1$
for $i_2 := 1$ to $n_2$
    $k := k + 1$
    .
    .
    .
for $i_m := 1$ to $n_m$
    $k := k + 1$
```

Solution: The initial value of k is zero. This block of code is made up of m different loops. Each time a loop is traversed, 1 is added to k. Let T_i be the task of traversing the ith loop. The task T_i can be done in n_i ways, since the ith loop is traversed n_i times. Since no two of these tasks can be done at the same time, the sum rule shows that the final value of k, which is the number of ways to do one of the tasks T_i, $i = 1, 2, \ldots, m$, is $n_1 + n_2 + \cdots + n_m$. ■

The sum rule can be phrased in terms of sets as follows: If $A_1, A_2, \ldots, A_m$ are disjoint sets, then the number of elements in the union of these sets is the sum of the numbers of elements in them. To relate this to our statement of the sum rule, let T_i be the task of choosing an element from A_i for $i = 1, 2, \ldots, m$. There are $|A_i|$ ways to do T_i. From the sum rule, since no two of the tasks can be done at the same time, the number of ways to choose an element from one of the sets, which is the number of elements in the union, is

$$|A_1 \cup A_2 \cup \cdots \cup A_m| = |A_1| + |A_2| + \cdots + |A_m|.$$

This equality applies only when the sets in question are disjoint. The situation is much more complicated when these sets have elements in common. That situation will be briefly discussed later in this section and discussed in more depth in Chapter 5.

The product rule applies when a procedure is made up of separate tasks.

THE PRODUCT RULE Suppose that a procedure can be broken down into two tasks. If there are n_1 ways to do the first task and n_2 ways to do the second task after the first task has been done, then there are $n_1 n_2$ ways to do the procedure.

The following examples show how the product rule is used.

EXAMPLE 4 The chairs of an auditorium are to be labeled with a letter and a positive integer not exceeding 100. What is the largest number of chairs that can be labeled differently?

Solution: The procedure of labeling a chair consists of two tasks, namely, assigning one of the 26 letters and then assigning one of the 100 possible integers to the seat. The product rule shows that there are $26 \cdot 100 = 2600$ different ways that a chair can be labeled. Therefore, the largest number of chairs that can be labeled differently is 2600. ■

EXAMPLE 5 There are 32 microcomputers in a computer center. Each microcomputer has 24 ports. How many different ports to a microcomputer in the center are there?

Solution: The procedure of choosing a port consists of two tasks, first picking a microcomputer and then picking a port on this microcomputer. Since there are 32 ways to choose the microcomputer and 24 ways to choose the port no matter which microcomputer has been selected, the product rule shows that there are 768 ports. ■

An extended version of the product rule is often useful. Suppose that a procedure is carried out by performing the tasks $T_1, T_2, \ldots, T_m$. If task T_i can be done in n_i ways after tasks $T_1, T_2, \ldots,$ and T_{i-1} have been done, then there are $n_1 \cdot n_2 \cdot \cdots \cdot n_m$ ways to carry out the procedure. This version of the product rule can be proved by mathematical induction from the product rule for two tasks (see Exercise 54 at the end of the section).

EXAMPLE 6 How many different bit strings are there of length seven?

Solution: Each of the seven bits can be chosen in two ways, since each bit is either zero or one. Therefore, the product rule shows there are a total of $2^7 = 128$ different bit strings of length seven. ■

EXAMPLE 7 How many different license plates are available if each plate contains a sequence of three letters followed by three digits (and no sequences of letters are prohibited, even if they are obscene)?

Solution: There are 26 choices for each of the three letters and 10 choices for each of the three digits. Hence, by the product rule there are a total of $26 \cdot 26 \cdot 26 \cdot 10 \cdot 10 \cdot 10 = 17{,}576{,}000$ possible license plates. ■

EXAMPLE 8 Counting Functions How many functions are there from a set with m elements to one with n elements?

Solution: A function corresponds to a choice of one of the n elements in the codomain for each of the m elements in the domain. Hence, by the product rule there are $n \cdot n \cdot \cdots \cdot n = n^m$ functions from a set with m elements to one with n elements. ■

EXAMPLE 9 Counting One-to-One Functions How many one-to-one functions are there from a set with m elements to one with n elements?

Solution: First note when $m > n$ there are no one-to-one functions from a set with m elements to a set with n elements. Now let $m \leq n$. Suppose the elements in the domain are $a_1, a_2, \ldots, a_m$. There are n ways to choose the value of the function at a_1. Since the function is one-to-one, the value of the function at a_2 can be picked in $n - 1$ ways (since the value used for a_1 cannot be used again). In general, the value of the function at a_k can be chosen in $n - k + 1$ ways. By the product rule, there are $n(n-1)(n-2)\cdots(n-m+1)$ one-to-one functions from a set with m elements to one with n elements. ■

EXAMPLE 10 The Telephone Numbering Plan The format of telephone numbers in North America is specified by a *numbering plan.* A telephone number consists of 10 digits, which are split into a three-digit area code, a three-digit office code, and a four-digit station code. Because of signaling considerations, there are certain restrictions on some of these digits. To specify the allowable format, let X denote a digit that can take any of the values 0 through 9, let N denote a digit that can take any of the values 2 through 9, and let Y denote a digit that must be a 0 or a 1. Two numbering plans, which will be called the old plan and the new plan, will be discussed. (The old plan, in use in the 1960s, has been replaced by the new plan, but the recent rapid growth in demand for new numbers will make even this new plan obsolete.) As will be shown, the new plan allows the use of more numbers.

In the old plan, the formats of the area code, office code, and station code are NYX, NNX, and $XXXX$, respectively. In the new plan, the formats of these codes are NXX, NXX, and $XXXX$, respectively. How many different North American telephone numbers are possible under the old plan and under the new plan?

Solution: By the product rule, there are $8 \cdot 2 \cdot 10 = 160$ area codes with format NYX and $8 \cdot 10 \cdot 10 = 800$ area codes with format NXX. Similarly, by the product rule, there are $8 \cdot 8 \cdot 10 = 640$ office codes with format NNX and $8 \cdot 10 \cdot 10 = 800$ with format NXX. The product rule also shows that there are $10 \cdot 10 \cdot 10 \cdot 10 = 10{,}000$ station codes with format $XXXX$.

Consequently, applying the product rule again, it follows that under the old plan there are

$$160 \cdot 640 \cdot 10{,}000 = 1{,}024{,}000{,}000$$

different numbers available in North America. Under the new plan there are

$$800 \cdot 800 \cdot 10{,}000 = 6{,}400{,}000{,}000$$

different numbers available. ∎

EXAMPLE 11 What is the value of k after the following code has been executed?

```
$k := 0$
for $i_1 := 1$ to $n_1$
  for $i_2 := 1$ to $n_2$
      .
      .
      .
  for $i_m := 1$ to $n_m$
      $k := k + 1$
```

Solution: The initial value of k is zero. Each time the nested loop is traversed, 1 is added to k. Let T_i be the task of traversing the ith loop. Then the number of times the loop is traversed is the number of ways to do the tasks $T_1, T_2, \ldots, T_m$. The number of ways to carry out the task T_j, $j = 1, 2, \ldots, m$, is n_j, since the jth loop is traversed once for each integer i_j with $1 \leq i_j \leq n_j$. By the product rule, it follows that the nested loop is traversed $n_1 n_2 \cdots n_m$ times. Hence, the final value of k is $n_1 n_2 \cdots n_m$. ∎

EXAMPLE 12 Counting Subsets of a Finite Set Use the product rule to show that the number of different subsets of a finite set S is $2^{|S|}$.

Solution: Let S be a finite set. List the elements of S in arbitrary order. Recall that there is a one-to-one correspondence between subsets of S and bit strings of length $|S|$. Namely, a subset of S is associated with the bit string with a 1 in the ith position if the ith element in the list is in the subset, and a 0 in this position otherwise. By the product rule, there are $2^{|S|}$ bit strings of length $|S|$. Hence, $|P(S)| = 2^{|S|}$. ■

The product rule is often phrased in terms of sets in the following way: If $A_1, A_2, \ldots, A_m$ are finite sets, then the number of elements in the Cartesian product of these sets is the product of the number of elements in each set. To relate this to the product rule, note that the task of choosing an element in the Cartesian product $A_1 \times A_2 \times \cdots \times A_m$ is done by choosing an element in A_1, an element in $A_2, \ldots,$ and an element in A_m. From the product rule it follows that

$$|A_1 \times A_2 \times \cdots \times A_m| = |A_1| \cdot |A_2| \cdot \cdots \cdot |A_m|.$$

MORE COMPLEX COUNTING PROBLEMS Many counting problems cannot be solved using just the sum rule or just the product rule. However, many complicated counting problems can be solved using both of these rules.

EXAMPLE 13 In a version of the computer language BASIC, the name of a variable is a string of one or two alphanumeric characters, where uppercase and lowercase letters are not distinguished. (An *alphanumeric* character is either one of the 26 English letters or one of the 10 digits.) Moreover, a variable name must begin with a letter and must be different from the five strings of two characters that are reserved for programming use. How many different variable names are there in this version of BASIC?

Solution: Let V equal the number of different variable names in this version of BASIC. Let V_1 be the number of these that are one character long and V_2 be the number of these that are two characters long. Then by the sum rule, $V = V_1 + V_2$. Note that $V_1 = 26$, since a one-character variable name must be a letter. Furthermore, by the product rule there are $26 \cdot 36$ strings of length two that begin with a letter and end with an alphanumeric character. However, five of these are excluded, so that $V_2 = 26 \cdot 36 - 5 = 931$. Hence, there are $V = V_1 + V_2 = 26 + 931 = 957$ different names for variables in this version of BASIC. ■

EXAMPLE 14 Each user on a computer system has a password, which is six to eight characters long, where each character is an uppercase letter or a digit. Each password must contain at least one digit. How many possible passwords are there?

Solution: Let P be the total number of possible passwords, and let P_6, P_7, and P_8 denote the number of possible passwords of length 6, 7, and 8, respectively. By the sum rule, $P = P_6 + P_7 + P_8$. We will now find P_6, P_7, and P_8. Finding P_6 directly is difficult. To find P_6 it is easier to find the number of strings of uppercase letters and digits that are six characters long, including those with no digits, and subtract from this the number of strings with no digits. By the product rule, the number of strings of six characters is 36^6, and the number of strings with no digits is 26^6. Hence,

$$P_6 = 36^6 - 26^6 = 2{,}176{,}782{,}336 - 308{,}915{,}776 = 1{,}867{,}866{,}560.$$

Similarly, it can be shown that

$$P_7 = 36^7 - 26^7 = 78{,}364{,}164{,}096 - 8{,}031{,}810{,}176 = 70{,}332{,}353{,}920$$

and

$$\begin{aligned} P_8 &= 36^8 - 26^8 = 2{,}821{,}109{,}907{,}456 - 208{,}827{,}064{,}576 \\ &= 2{,}612{,}282{,}842{,}880. \end{aligned}$$

Consequently,

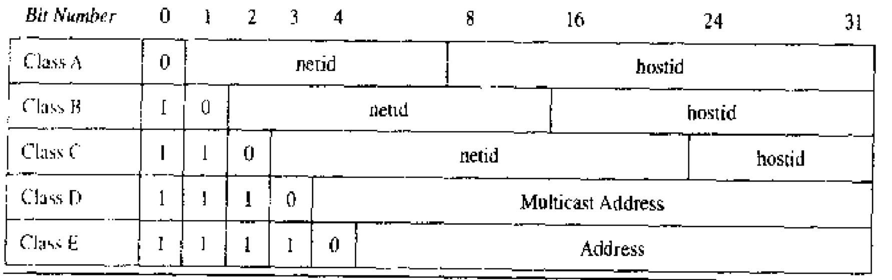
$$P = P_6 + P_7 + P_8 = 2{,}684{,}483{,}063{,}360.$$ ■

EXAMPLE 15 Counting Internet Addresses. In the Internet, which is made up of interconnected physical networks of computers, each computer (or more precisely, each network connection of a computer) is assigned an *Internet address*. In Version 4 of the Internet Protocol (IPv4), now in use, an address is a string of 32 bits. It begins with a *network number* (*netid*). The netid is followed by a *host number* (*hostid*), which identifies a computer as a member of a particular network.

web

Three forms of addresses are used, with different numbers of bits used for netids and hostids. **Class A addresses,** used for the largest networks, consist of 0, followed by a 7-bit netid and a 24-bit hostid. **Class B addresses,** used for medium-sized networks, consist of 10, followed by a 14-bit netid and a 16-bit hostid. **Class C addresses,** used for the smallest networks, consist of 110, followed by a 21-bit netid and an 8-bit hostid. There are several restrictions on addresses because of special uses: 1111111 is not available as the netid of a Class A network, and the hostids consisting of all 0s and all 1s are not available for use in any network. A computer on the Internet has either a Class A, a Class B, or a Class C address. (Besides Class A, B, and C addresses, there are also Class D addresses, reserved for use in multicasting when multiple computers are addressed at a single time, consisting of 1110 followed by 28 bits, and Class E addresses, reserved for future use, consisting of 11110 followed by 27 bits. Neither Class D nor Class E addresses are assigned as the IP address of a computer on the Internet.) Figure 1 illustrates IPv4 addressing. (Limitations on the number of Class A and Class B netids have made IPv4 addressing inadequate; IPv6, which will replace IPv4, uses 128-bit addresses to solve this problem.)

How many different IPv4 addresses are available for computers on the Internet?



Class A	0	netid	hostid			
Class B	1	0	netid	hostid		
Class C	1	1	0	netid	hostid	
Class D	1	1	1	0	Multicast Address	
Class E	1	1	1	1	0	Address

FIGURE 1 Internet Addresses (IPv4).

Solution: Let x be the number of available addresses for computers on the Internet, and let x_A, x_B, and x_C denote the number of Class A, Class B, and Class C addresses available, respectively. By the sum rule, $x = x_A + x_B + x_C$.

To find x_A, note that there are $2^7 - 1 = 127$ Class A netids, recalling that the netid 1111111 is unavailable. For each netid, there are $2^{24} - 2 = 16{,}777{,}214$ hostids, recalling that the hostids consisting of all 0s and all 1s are unavailable. Consequently, $x_A = 127 \cdot 16{,}777{,}214 = 2{,}130{,}706{,}178$.

To find x_B and x_C, note that there are $2^{14} = 16{,}384$ Class B netids and $2^{21} = 2{,}097{,}152$ Class C netids. For each Class B netid, there are $2^{16} - 2 = 65{,}534$ hostids, and for each Class C netid, there are $2^8 - 2 = 254$ hostids, recalling that in each network the hostids consisting of all 0s and all 1s are unavailable. Consequently, $x_B = 1{,}073{,}709{,}056$ and $x_C = 532{,}676{,}608$.

We conclude that the total number of IPv4 addresses available is $x = x_A + x_B + x_C = 2{,}130{,}706{,}178 + 1{,}073{,}709{,}056 + 532{,}676{,}608 = 3{,}737{,}091{,}842$. ■

THE INCLUSION–EXCLUSION PRINCIPLE

When two tasks can be done at the same time, we cannot use the sum rule to count the number of ways to do one of the two tasks. Adding the number of ways to do each task leads to an overcount, since the ways to do both tasks are counted twice. To correctly count the number of ways to do one of the two tasks, we add the number of ways to do each of the two tasks and then subtract the number of ways to do both tasks. This technique is called the **principle of inclusion–exclusion.** Example 16 illustrates how we can solve counting problems using this principle.

EXAMPLE 16 How many bit strings of length eight either start with a 1 bit or end with the two bits 00?

Solution: The first task, constructing a bit string of length eight beginning with a 1 bit, can be done in $2^7 = 128$ ways. This follows by the product rule, since the first bit can be chosen in only one way and each of the other seven bits can be chosen in two ways.

The second task, constructing a bit string of length eight ending with the two bits 00, can be done in $2^6 = 64$ ways. This follows by the product rule, since each of the first six bits can be chosen in two ways and the last two bits can be chosen in only one way.

Both tasks, constructing a bit string of length eight that begins with a 1 and ends with 00, can be done in $2^5 = 32$ ways. This follows by the product rule, since the first bit can be chosen in only one way, each of the second through the sixth bits can be chosen in two ways, and the last two bits can be chosen in one way. Consequently, the number of bit strings of length eight that begin with a 1 and end with a 00, which equals the number of ways to do either the first task or the second task, equals $128 + 64 - 32 = 160$. ■

We can phrase this counting principle in terms of sets. Let A_1 and A_2 be sets. Let T_1 be the task of choosing an element from A_1 and T_2 the task of choosing an element from A_2. There are $|A_1|$ ways to do T_1 and $|A_2|$ ways to do T_2. The number of ways to do either T_1 or T_2 is the sum of the number of ways to do T_1 and the number of ways to do T_2, minus the number of ways to do both T_1 and T_2. Since there are $|A_1 \cup A_2|$ ways to do either T_1 or T_2 and $|A_1 \cap A_2|$ ways to do both T_1 and T_2, we have

$$|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2|.$$

This is the formula given in Section 1.5 for the number of elements in the union of two sets.

The principle of inclusion–exclusion can be generalized to find the number of ways to do one of *n* different tasks or, equivalently, to find the number of elements in the union of *n* sets, whenever *n* is a positive integer. We will study the inclusion–exclusion principle and some of its many applications in Chapter 5.

TREE DIAGRAMS

Counting problems can be solved using **tree diagrams.** A tree consists of a root, a number of branches leaving the root, and possible additional branches leaving the endpoints of other branches. (We will study trees in detail in Chapter 8.) To use trees in counting, we use a branch to represent each possible choice. We represent the possible outcomes by the leaves, which are the endpoints of branches not having other branches starting at them.

EXAMPLE 17 How many bit strings of length four do not have two consecutive 1s?

Solution: The tree diagram in Figure 2 displays all bit strings of length four without two consecutive 1s. We see that there are eight bit strings of length four without two consecutive 1s. ■

EXAMPLE 18 A playoff between two teams consists of at most five games. The first team that wins three games wins the playoff. In how many different ways can the playoff occur?

Solution: The tree diagram in Figure 3 displays all the ways the playoff can proceed, with the winner of each game shown. We see that there are 20 different ways for the playoff to occur. ■

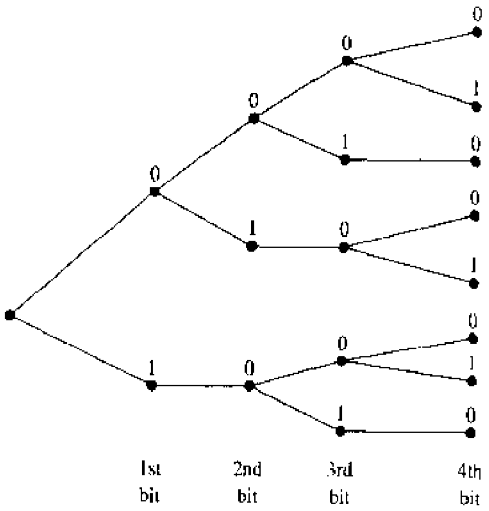


FIGURE 2 **Bit Strings of Length Four Without Consecutive 1s.**

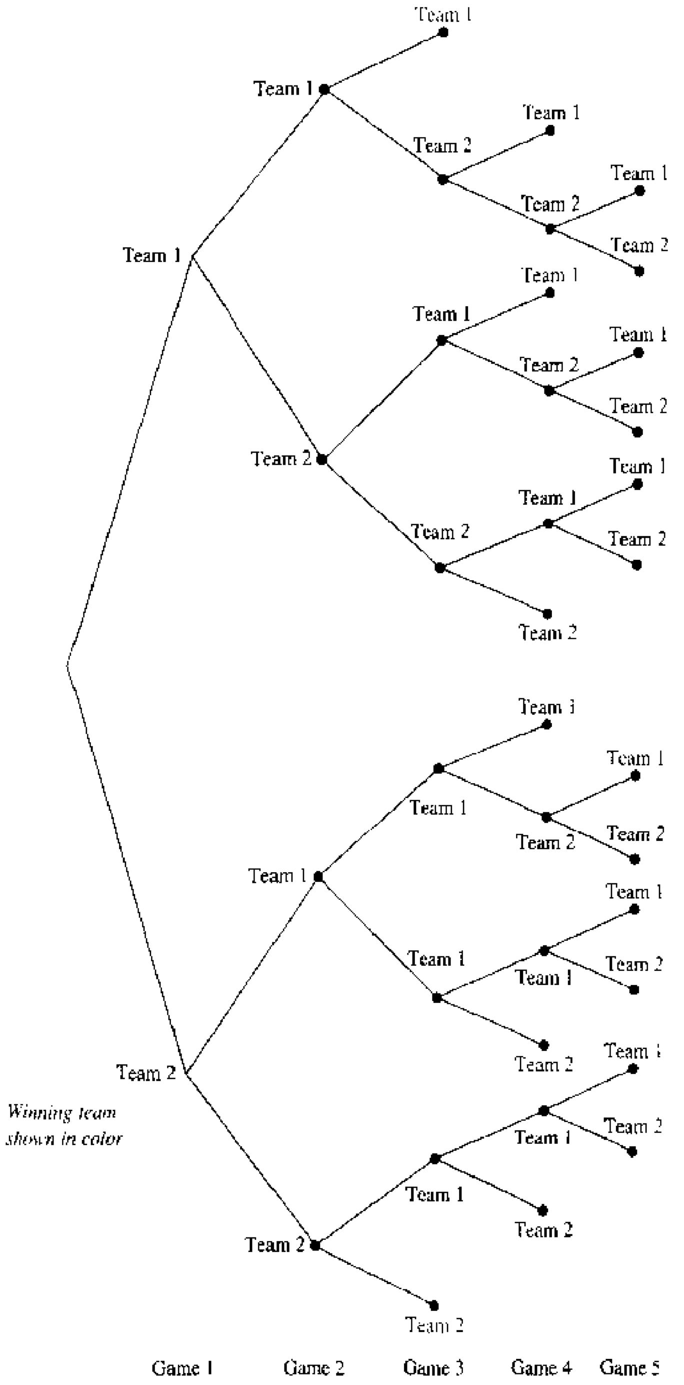

Team 1
Team 2
Game 1
Game 2
Game 3
Game 4
Game 5
Winning team shown in color

FIGURE 3 **Best Three Games Out of Five Playoffs.**

Exercises

1. There are 18 mathematics majors and 325 computer science majors at a college.
   a) How many ways are there to pick two representatives, so that one is a mathematics major and the other is a computer science major?
   b) How many ways are there to pick one representative who is either a mathematics major or a computer science major?
2. An office building contains 27 floors and has 37 offices on each floor. How many offices are in the building?
3. A multiple-choice test contains 10 questions. There are four possible answers for each question.
   a) How many ways can a student answer the questions on the test if every question is answered?
   b) How many ways can a student answer the questions on the test if the student can leave answers blank?
4. A particular brand of shirt comes in 12 colors, has a male version and a female version, and comes in three sizes for each sex. How many different types of this shirt are made?
5. There are six different airlines that fly from New York to Denver and seven that fly from Denver to San Francisco. How many different possibilities are there for a trip from New York to San Francisco via Denver, when an airline is picked for the flight to Denver and an airline is picked for the continuation flight to San Francisco?
6. There are four major auto routes from Boston to Detroit and six from Detroit to Los Angeles. How many major auto routes are there from Boston to Los Angeles via Detroit?
7. How many different three-letter initials can people have?
8. How many different three-letter initials with none of the letters repeated can people have?
9. How many different three-letter initials are there that begin with an A?
10. How many bit strings are there of length eight?
11. How many bit strings of length 10 begin and end with a 1?
12. How many bit strings are there of length six or less?
13. How many bit strings with length not exceeding n, where n is a positive integer, consist entirely of 1s?
14. How many bit strings of length n, where n is a positive integer, start and end with 1s?
15. How many strings are there of lowercase letters of length four or less?
16. How many strings are there of four lowercase letters that have the letter x in them?
17. How many strings of five ASCII characters contain the character @ (at sign) at least once? (*Note:* There are 128 different ASCII characters.)
18. How many positive integers less than 1000 are
   a) divisible by 7?
   b) divisible by 7 but not by 11?
   c) divisible by both 7 and 11?
   d) divisible by either 7 or 11?
   e) divisible by exactly one of 7 and 11?
   f) divisible by neither 7 nor 11?
   g) have distinct digits?
   h) have distinct digits and are even?
19. How many positive integers with exactly three decimal digits, that is, positive integers between 100 and 999 inclusive,
   a) are divisible by 7?
   b) are odd?
   c) have the same three decimal digits?
   d) are not divisible by 4?
   e) are divisible by 3 or 4?
   f) are not divisible by either 3 or 4?
   g) are divisible by 3 but not by 4?
   h) are divisible by 3 and 4?
20. How many positive integers with exactly four decimal digits, that is, positive integers between 1000 and 9999 inclusive,
   a) are divisible by 9?
   b) are even?
   c) have distinct digits?
   d) are not divisible by 3?
   e) are divisible by 5 or 7?
   f) are not divisible by either 5 or 7?
   g) are divisible by 5 but not by 7?
   h) are divisible by 5 and 7?
21. How many strings of three decimal digits
   a) do not contain the same digit three times?
   b) begin with an odd digit?
   c) have exactly two digits that are 4s?
22. How many strings of four decimal digits
   a) do not contain the same digit twice?
   b) end with an even digit?
   c) have exactly three digits that are 9s?
23. A committee is formed containing either the governor or one of the two senators of each of the 50 states. How many ways are there to form this committee?
24. How many license plates can be made using either three digits followed by three letters or three letters followed by three digits?
25. How many license plates can be made using either two letters followed by four digits or two digits followed by four letters?
26. How many license plates can be made using either three letters followed by three digits or four letters followed by two digits?
27. How many license plates can be made using either two or three letters followed by either two or three digits?

28. How many strings of eight English letters are there
- a) if letters can be repeated?
- b) if no letter can be repeated?
- c) that start with X, if letters can be repeated?
- d) that start with X, if no letter can be repeated?
- e) that start and end with X, if letters can be repeated?
- f) that start with the letters BO (in that order), if letters can be repeated?
- g) that start and end with the letters BO (in that order), if letters can be repeated?
- h) that start or end with the letters BO (in that order), if letters can be repeated?

29. How many strings of eight English letters are there
- a) that contain no vowels, if letters can be repeated?
- b) that contain no vowels, if letters cannot be repeated?
- c) that start with a vowel, if letters can be repeated?
- d) that start with a vowel, if letters cannot be repeated?
- e) that contain at least one vowel, if letters can be repeated?
- f) that contain exactly one vowel, if letters can be repeated?
- g) that start with X and contain at least one vowel, if letters can be repeated?
- h) that start and end with X and contain at least one vowel, if letters can be repeated?

30. How many different functions are there from a set with 10 elements to sets with the following numbers of elements?

a) 2   b) 3   c) 4   d) 5

31. How many one-to-one functions are there from a set with five elements to sets with the following number of elements?

a) 4   b) 5   c) 6   d) 7

32. How many functions are there from the set $\{1, 2, \ldots, n\}$, where n is a positive integer, to the set $\{0, 1\}$?

33. How many functions are there from the set $\{1, 2, \ldots, n\}$, where n is a positive integer, to the set $\{0, 1\}$
- a) that are one-to-one?
- b) that assign 0 to both 1 and n?
- c) that assign 1 to exactly one of the positive integers less than n?

34. How many partial functions (see the exercises in Section 1.6) are there from a set with five elements to sets with the following number of elements?

a) 1   b) 2   c) 5   d) 9

35. How many partial functions (see the exercises in Section 1.6) are there from a set with m elements to a set with n elements, where m and n are positive integers?

36. How many subsets of a set with 100 elements have more than one element?

37. A **palindrome** is a string whose reversal is identical to the string. How many bit strings of length n are palindromes?

38. In how many ways can a photographer at a wedding arrange 6 people in a row from a group of 10 people, where the bride and the groom are among these 10 people, if
- a) the bride must be in the picture?
- b) both the bride and groom must be in the picture?
- c) exactly one of the bride and the groom is in the picture?

39. In how many ways can a photographer at a wedding arrange six people in a row, including the bride and groom, if
- a) the bride must be next to the groom?
- b) the bride is not next to the groom?
- c) the bride is positioned somewhere to the left of the groom?

40. How many bit strings of length seven either begin with two 0s or end with three 1s?

41. How many bit strings of length 10 either begin with three 0s or end with two 0s?

***42.** How many bit strings of length 10 contain either five consecutive 0s or five consecutive 1s?

****43.** How many bit strings of length eight contain either three consecutive 0s or four consecutive 1s?

44. Every student in a discrete mathematics class is either a computer science or a mathematics major or is a joint major in these two subjects. How many students are in the class if there are 38 computer science majors (including joint majors), 23 mathematics majors (including joint majors), and 7 joint majors?

45. How many positive integers not exceeding 100 are divisible either by 4 or by 6?

46. The name of a variable in the C programming language is a string that can contain uppercase letters, lowercase letters, digits, or underscores. Further, the first character in the string must be a letter, either uppercase or lowercase, or an underscore. If the name of a variable is determined by its first eight characters, how many different variables can be named in C? (Note that the name of a variable may contain fewer than eight characters.)

47. Suppose that at some future time every telephone in the world is assigned a number that contains a country code 1 to 3 digits long, that is, of the form X, XX, or XXX, followed by a 10-digit telephone number of the form NXX-NXX-$XXXX$ (as described in Example 10). How many different telephone numbers would be available worldwide under this numbering plan?

48. Use a tree diagram to find the number of bit strings of length four with no three consecutive 0s.

49. How many ways are there to arrange the letters a, b, c, and d such that a is not followed immediately by b?

50. Use a tree diagram to find the number of ways that the World Series can occur, where the first team that wins four games out of seven wins the series.

51. Use a tree diagram to determine the number of subsets of $\{3, 7, 9, 11, 24\}$ with the property that the sum of the elements in the subset is less than 28.

***52.** Use the product rule to show that there are 2^{2^n} different truth tables for propositions in n variables.

53. Use mathematical induction to prove the sum rule for m tasks from the sum rule for two tasks.

54. Use mathematical induction to prove the product rule for m tasks from the product rule for two tasks.

55. How many diagonals does a convex polygon with n sides have? (A polygon is convex if every line segment connecting two points in the interior or boundary of the polygon lies entirely within this set.)

56. Data are transmitted over the Internet in **datagrams**, which are structured blocks of bits. Each datagram contains header information organized into a maximum of 14 different fields (specifying many things, including the source and destination addresses) and a data area that contains the actual data which are transmitted. One of the 14 header fields is the **header length field** (denoted by HLEN), which is specified by the protocol to be 4 bits long and which specifies the header length in terms of 32-bit blocks of bits. For example, if HLEN = 0110, the header is made up of six 32-bit blocks. Another of the 14 header fields is the 16-bit-long **total length field** (denoted by TOTAL LENGTH), which specifies the length in bits of the entire datagram, including both the header fields and the data area. The length of the data area is the total length of the datagram minus the length of the header.

a) The largest possible value of TOTAL LENGTH (which is 16 bits long) determines the maximum total length in octets (blocks of 8 bits) of an Internet datagram. What is this value?

b) The largest possible value of HLEN (which is 4 bits long) determines the maximum total header length in 32-bit blocks. What is this value? What is the maximum total header length in octets?

c) The minimum (and most common) header length is 20 octets. What is the maximum total length in octets of the data area of an Internet datagram?

d) How many different strings of octets in the data area can be transmitted if the header length is 20 octets and the total length is as long as possible?

4.2

The Pigeonhole Principle

INTRODUCTION

Suppose that a flock of pigeons flies into a set of pigeonholes to roost. The **pigeonhole principle** states that if there are more pigeons than pigeonholes, then there must be at least one pigeonhole with at least two pigeons in it (see Figure 1). Of course, this principle applies to other objects besides pigeons and pigeonholes.

THEOREM 1 THE PIGEONHOLE PRINCIPLE **If $k + 1$ or more objects are placed into k boxes, then there is at least one box containing two or more of the objects.**

Proof: Suppose that none of the k boxes contains more than one object. Then the total number of objects would be at most k. This is a contradiction, since there are at least $k + 1$ objects. □

The pigeonhole principle is also called the **Dirichlet drawer principle,** after the nineteenth-century German mathematician Dirichlet, who often used this principle in his work. The following examples show how the pigeonhole principle is used.

G. Lejeune Dirichlet (1805–1859). G. Lejeune Dirichlet was born into a French family living near Cologne, Germany. He studied at the University of Paris and held positions at the University of Breslau and the University of Berlin. In 1855 he was chosen to succeed Gauss at the University of Göttingen. Dirichlet is said to be the first person to master Gauss's *Disquisitiones Arithmeticae*, which appeared 20 years earlier. He is said to have kept a copy at his side even when he traveled. Dirichlet made many important discoveries in number theory, including the theorem that there are infinitely many primes in arithmetical progressions $an + b$ when a and b are relatively prime. He proved the $n = 5$ case of Fermat's Last Theorem, that there are no nontrivial solutions in integers to $x^5 + y^5 = z^5$. Dirichlet also made many contributions to analysis.

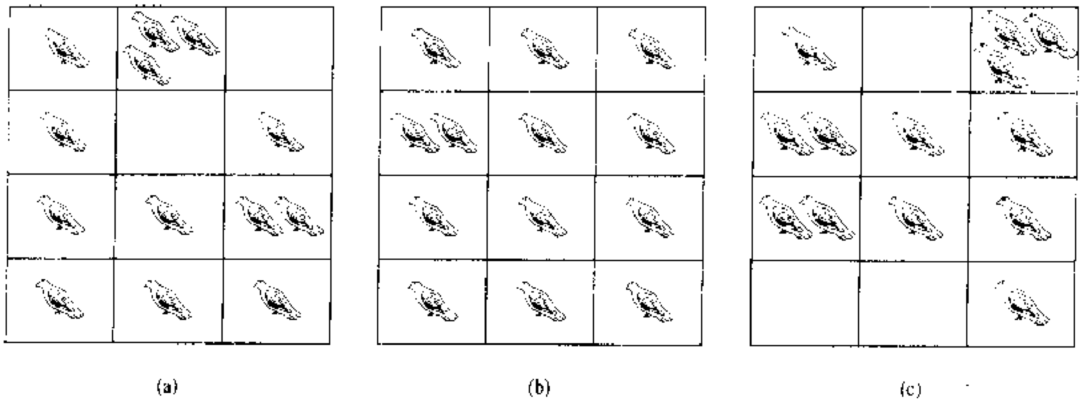

(a) (b) (c)

FIGURE 1 There Are More Pigeons Than Pigeonholes.

EXAMPLE 1 Among any group of 367 people, there must be at least two with the same birthday, because there are only 366 possible birthdays. ■

EXAMPLE 2 In any group of 27 English words, there must be at least two that begin with the same letter, since there are 26 letters in the English alphabet. ■

EXAMPLE 3 How many students must be in a class to guarantee that at least two students receive the same score on the final exam, if the exam is graded on a scale from 0 to 100 points?

Solution: There are 101 possible scores on the final. The pigeonhole principle shows that among any 102 students there must be at least 2 students with the same score. ■

THE GENERALIZED PIGEONHOLE PRINCIPLE

The pigeonhole principle states that there must be at least two objects in the same box when there are more objects than boxes. However, even more can be said when the number of objects exceeds a multiple of the number of boxes. For instance, among any set of 21 decimal digits there must be 3 that are the same. This follows because when 21 objects are distributed into 10 boxes, one box must have more than 2 objects.

THEOREM 2 **THE GENERALIZED PIGEONHOLE PRINCIPLE If N objects are placed into k boxes, then there is at least one box containing at least $\lceil N/k \rceil$ objects.**

Proof: Suppose that none of the boxes contains more than $\lceil N/k \rceil - 1$ objects. Then, the total number of objects is at most

$$k(\lceil N/k \rceil - 1) < k(((N/k) + 1) - 1) = N,$$

where the inequality $\lceil N/k \rceil < (N/k) + 1$ has been used. This is a contradiction since there are a total of N objects. □

The following examples illustrate how the generalized pigeonhole principle is applied.

EXAMPLE 4 Among 100 people there are at least $\lceil 100/12 \rceil = 9$ who were born in the same month. ■

EXAMPLE 5 What is the minimum number of students required in a discrete mathematics class to be sure that at least six will receive the same grade, if there are five possible grades, A, B, C, D, and F?

Solution: The minimum number of students needed to guarantee that at least six students receive the same grade is the smallest integer N such that $\lceil N/5 \rceil = 6$. The smallest such integer is $N = 5 \cdot 5 + 1 = 26$. Thus, 26 is the minimum number of students needed to be sure that at least 6 students will receive the same grade. ■

EXAMPLE 6 What is the least number of area codes needed to guarantee that the 25 million phones in a state have distinct 10-digit telephone numbers? (Assume that telephone numbers are of the form *NXX*–*NXX*–*XXXX*, where the first three digits form the area code, N represents a digit from 2 to 9 inclusive, and X represents any digit.)

Solution: There are 8 million different phone numbers of the form *NXX*–*XXXX* (as shown in Example 10 of Section 4.1). Hence, by the generalized pigeonhole principle, among 25 million telephones, at least $\lceil 25{,}000{,}000/8{,}000{,}000 \rceil$ of them must have identical phone numbers. Hence, at least four area codes are required to ensure that all 10-digit numbers are different. ■

SOME ELEGANT APPLICATIONS OF THE PIGEONHOLE PRINCIPLE

In many interesting applications of the pigeonhole principle, the objects to be placed in boxes must be chosen in a clever way. A few such applications will be described here.

EXAMPLE 7 During a month with 30 days a baseball team plays at least 1 game a day, but no more than 45 games. Show that there must be a period of some number of consecutive days during which the team must play exactly 14 games.

Solution: Let a_j be the number of games played on or before the jth day of the month. Then $a_1, a_2, \ldots, a_{30}$ is an increasing sequence of distinct positive integers, with $1 \le a_j \le 45$. Moreover, $a_1 + 14, a_2 + 14, \ldots, a_{30} + 14$ is also an increasing sequence of distinct positive integers, with $15 \le a_j + 14 \le 59$.

The 60 positive integers $a_1, a_2, \ldots, a_{30}, a_1 + 14, a_2 + 14, \ldots, a_{30} + 14$ are all less than or equal to 59. Hence, by the pigeonhole principle two of these integers are equal. Since the integers a_j, $j = 1, 2, \ldots, 30$ are all distinct and the integers $a_j + 14$, $j = 1, 2, \ldots, 30$ are all distinct, there must be indices i and j with $a_i = a_j + 14$. This means that exactly 14 games were played from day $j + 1$ to day i. ■

EXAMPLE 8 Show that among any $n + 1$ positive integers not exceeding $2n$ there must be an integer that divides one of the other integers.

Solution: Write each of the $n+1$ integers $a_1, a_2, \ldots, a_{n+1}$ as a power of 2 times an odd integer. In other words, let $a_j = 2^{k_j}q_j$ for $j = 1, 2, \ldots, n+1$, where k_j is a nonnegative integer and q_j is odd. The integers $q_1, q_2, \ldots, q_{n+1}$ are all odd positive integers less than $2n$. Since there are only n odd positive integers less than $2n$, it follows from the pigeonhole principle that two of the integers $q_1, q_2, \ldots, q_{n+1}$ must be equal. Therefore, there are integers i and j such that $q_i = q_j$. Let q be the common value of q_i and q_j. Then, $a_i = 2^{k_i}q$ and $a_j = 2^{k_j}q$. It follows that if $k_i < k_j$, then a_i divides a_j; while if $k_i > k_j$, then a_j divides a_i. ■

A clever application of the pigeonhole principle shows the existence of an increasing or a decreasing subsequence of a certain length in a sequence of distinct integers. Some definitions will be reviewed before this application is presented. Suppose that $a_1, a_2, \ldots, a_N$ is a sequence of real numbers. A **subsequence** of this sequence is a sequence of the form $a_{i_1}, a_{i_2}, \ldots, a_{i_m}$, where $1 \le i_1 < i_2 < \cdots < i_m \le N$. Hence, a subsequence is a sequence obtained from the original sequence by including some of the terms of the original sequence in their original order, and perhaps not including other terms. A sequence is called **strictly increasing** if each term is larger than the one that precedes it, and it is called **strictly decreasing** if each term is smaller than the one that precedes it.

THEOREM 3 **Every sequence of $n^2 + 1$ distinct real numbers contains a subsequence of length $n + 1$ that is either strictly increasing or strictly decreasing.**

The following example will be given before the theorem is proved.

EXAMPLE 9 The sequence 8, 11, 9, 1, 4, 6, 12, 10, 5, 7 contains 10 terms. Note that $10 = 3^2 + 1$. There are four increasing subsequences of length four, namely, 1, 4, 6, 12; 1, 4, 6, 7; 1, 4, 6, 10; and 1, 4, 5, 7. There is also a decreasing subsequence of length four, namely, 11, 9, 6, 5. ■

The proof of the theorem will now be given.

Proof: Let $a_1, a_2, \ldots, a_{n^2+1}$ be a sequence of $n^2 + 1$ distinct real numbers. Associate an ordered pair with each term of the sequence, namely, associate (i_k, d_k) to the term a_k, where i_k is the length of the longest increasing subsequence starting at a_k, and d_k is the length of the longest decreasing subsequence starting at a_k.

Suppose that there are no increasing or decreasing subsequences of length $n + 1$. Then i_k and d_k are both positive integers less than or equal to n, for $k = 1, 2, \ldots, n^2 + 1$. Hence, by the product rule there are n^2 possible ordered pairs for (i_k, d_k). By the pigeonhole principle, two of these $n^2 + 1$ ordered pairs are equal. In other words, there exist terms a_s and a_t, with $s < t$ such that $i_s = i_t$ and $d_s = d_t$. We will show that this is impossible. Because the terms of the sequence are distinct, either $a_s < a_t$ or $a_s > a_t$. If $a_s < a_t$, then, since $i_s = i_t$, an increasing subsequence of length $i_t + 1$ can be built starting at a_s, by taking a_s followed by an increasing subsequence of length i_t beginning at a_t. This is a contradiction. Similarly, if $a_s > a_t$, it can be shown that d_s must be greater than d_t, which is a contradiction. □

The final example shows how the generalized pigeonhole principle can be applied to an important part of combinatorics called **Ramsey theory,** after the English mathematician F. P. Ramsey. In general, Ramsey theory deals with the distribution of subsets of elements of sets.

EXAMPLE 10 Assume that in a group of six people, each pair of individuals consists of two friends or two enemies. Show that there are either three mutual friends or three mutual enemies in the group.

Solution: Let A be one of the six people. Of the five other people in the group, there are either three or more who are friends of A, or three or more who are enemies of A. This follows from the generalized pigeonhole principle, since when five objects are divided into two sets, one of the sets has at least $\lceil 5/2 \rceil = 3$ elements. In the former case, suppose that B, C, and D are friends of A. If any two of these three individuals are friends, then these two and A form a group of three mutual friends. Otherwise, B, C, and D form a set of three mutual enemies. The proof in the latter case, when there are three or more enemies of A, proceeds in a similar manner. ■

Exercises

1. Show that in any set of six classes there must be two that meet on the same day, assuming that no classes are held on weekends.
2. Show that if there are 30 students in a class, then at least 2 have last names that begin with the same letter.
3. A drawer contains a dozen brown socks and a dozen black socks, all unmatched. A man takes socks out at random in the dark.
   a) How many socks must he take out to be sure that he has at least two socks of the same color?
   b) How many socks must he take out to be sure that he has at least two black socks?
4. A bowl contains 10 red balls and 10 blue balls. A woman selects balls at random without looking at them.
   a) How many balls must she select to be sure of having at least three balls of the same color?
   b) How many balls must she select to be sure of having at least three blue balls?
5. Show that among any group of five (not necessarily consecutive) integers, there are two with the same remainder when divided by 4.
6. Let d be a positive integer. Show that among any group of $d + 1$ (not necessarily consecutive) integers there are two with exactly the same remainder when they are divided by d.
7. Let n be a positive integer. Show that in any set of n consecutive integers there is exactly one divisible by n.
8. Show that if f is a function from S to T where S and T are finite sets with $|S| > |T|$, then there are elements s_1 and s_2 in S such that $f(s_1) = f(s_2)$, or in other words, f is not one-to-one.
9. How many students, each of whom comes from one of the 50 states, must be enrolled in a university to guarantee that there are at least 100 who come from the same state?

Frank Plumpton Ramsey (1903–1930). Frank Plumpton Ramsey, son of the president of Magdalene College, Cambridge, was educated at Winchester and Trinity Colleges. After graduating in 1923, he was elected a fellow of King's College, Cambridge, where he spent the remainder of his life. Ramsey made important contributions to mathematical logic. What we now call Ramsey theory began with his clever combinatorial arguments, published in the paper "On a Problem of Formal Logic." Ramsey also made contributions to the mathematical theory of economics. He was noted as an excellent lecturer on the foundations of mathematics. His death at the age of 26 deprived the mathematical community and Cambridge University of a brilliant young scholar.

*10. Let (x_i, y_i), $i = 1, 2, 3, 4, 5$, be a set of five distinct points with integer coordinates in the xy plane. Show that the midpoint of the line joining at least one pair of these points has integer coordinates.

*11. Let (x_i, y_i, z_i), $i = 1, 2, 3, 4, 5, 6, 7, 8, 9$, be a set of nine distinct points with integer coordinates in xyz space. Show that the midpoint of at least one pair of these points has integer coordinates.

12. How many ordered pairs of integers (a, b) are needed to guarantee that there are two ordered pairs (a_1, b_1) and (a_2, b_2) such that $a_1 \textbf{ mod } 5 = a_2 \textbf{ mod } 5$ and $b_1 \textbf{ mod } 5 = b_2 \textbf{ mod } 5$?

13. a) Show that if five integers are selected from the first eight positive integers, there must be a pair of these integers with a sum equal to 9.
    b) Is the conclusion in part (a) true if four integers are selected rather than five?

14. a) Show that if seven integers are selected from the first 10 positive integers, there must be at least two pairs of these integers with the sum 11.
    b) Is the conclusion in part (a) true if six integers are selected rather than seven?

15. A company stores products in a warehouse. Storage bins in this warehouse are specified by their aisle, location in the aisle, and shelf. There are 50 aisles, 85 horizontal locations in each aisle, and 5 shelves throughout the warehouse. What is the least number of products the company can have so that at least two products must be stored in the same bin?

16. There are 51 houses on a street. Each house has an address between 1000 and 1099, inclusive. Show that at least two houses have addresses that are consecutive integers.

*17. Let x be an irrational number. Show that the absolute value of the difference between jx and the nearest integer to jx is less than $1/n$ for some positive integer j not exceeding n.

18. Find an increasing subsequence of maximal length and a decreasing subsequence of maximal length in the sequence 22, 5, 7, 2, 23, 10, 15, 21, 3, 17.

19. Construct a sequence of 16 positive integers that has no increasing or decreasing subsequence of 5 terms.

20. Show that if there are 101 people of different heights standing in a line, it is possible to find 11 people in the order they are standing in the line with heights that are either increasing or decreasing.

*21. Describe an algorithm in pseudocode for producing the largest increasing or decreasing subsequence of a sequence of distinct integers.

22. Show that in a group of five people (where any two people are either friends or enemies), there are not necessarily three mutual friends or three mutual enemies.

23. Show that in a group of 10 people (where any 2 people are either friends or enemies), there are either 3 mutual friends or 4 mutual enemies, and there are either 3 mutual enemies or 4 mutual friends.

24. Use Exercise 23 to show that among any group of 20 people (where any 2 people are either friends or enemies), there are either 4 mutual friends or 4 mutual enemies.

25. Show that there are at least four people in California (population: 25 million) with the same three initials who were born on the same day of the year (but not necessarily in the same year).

26. Show that if there are 100,000,000 wage earners in the United States who earn less than 1,000,000 dollars, then there are two who earned exactly the same amount of money, to the penny, last year.

27. There are 38 different time periods during which classes at a university can be scheduled. If there are 677 different classes, how many different rooms will be needed?

28. A computer network consists of six computers. Each computer is directly connected to at least one of the other computers. Show that there are at least two computers in the network that are directly connected to the same number of other computers.

29. A computer network consists of six computers. Each computer is directly connected to zero or more of the other computers. Show that there are at least two computers in the network that are directly connected to the same number of other computers.

*30. Prove that at a party where there are at least two people, there are two people who know the same number of other people there.

31. An arm wrestler is the champion for a period of 75 hours. The arm wrestler had at least one match an hour, but no more than 125 total matches. Show that there is a period of consecutive hours during which the arm wrestler had exactly 24 matches.

*32. Is the statement in Exercise 31 true if 24 is replaced by
    a) 2? b) 23? c) 25? d) 30?

33. Show that if f is a function from S to T where S and T are finite sets and $m = \lceil |S|/|T| \rceil$, then there are at least m elements of S that are mapped to the same value of T. That is, show that there are elements $s_1, s_2, \ldots, s_m$ of S such that $f(s_1) = f(s_2) = \cdots = f(s_m)$.

34. Suppose that there are nine students in a discrete mathematics class at a small college.
    a) Show that the class must have at least five male students or at least five female students.
    b) Show that the class must have at least three male students or at least seven female students.

35. Suppose that every student in a discrete mathematics class of 25 students is a freshman, a sophomore, or a junior.
    a) Show that there are at least 9 freshman, at least 9 sophomores, or at least 9 juniors in the class.

EXAMPLE 6 We see that $C(4, 2) = 6$, since the 2-combinations of $\{a, b, c, d\}$ are the six subsets $\{a, b\}$, $\{a, c\}$, $\{a, d\}$, $\{b, c\}$, $\{b, d\}$, and $\{c, d\}$. ■

We can determine the number of r-combinations of a set with n elements using the formula for the number of r-permutations of a set. To do this, note that the r-permutations of a set can be obtained by first forming r-combinations and then ordering the elements in these combinations. The proof of the following theorem, which gives the value of $C(n, r)$, is based on this observation.

THEOREM 2 **The number of r-combinations of a set with n elements, where n is a positive integer and r is an integer with $0 \le r \le n$, equals**

$$C(n, r) = \frac{n!}{r!\,(n-r)!}.$$

Proof: The r-permutations of the set can be obtained by forming the $C(n, r)$ r-combinations of the set, and then ordering the elements in each r-combination, which can be done in $P(r, r)$ ways. Consequently,

$$P(n, r) = C(n, r) \cdot P(r, r).$$

This implies that

$$C(n, r) = \frac{P(n, r)}{P(r, r)} = \frac{n!/(n-r)!}{r!/(r-r)!} = \frac{n!}{r!\,(n-r)!}. \qquad \square$$

The following corollary is helpful in computing the number of r-combinations of a set.

COROLLARY 1 **Let n and r be nonnegative integers with $r \le n$. Then $C(n, r) = C(n, n-r)$.**

Proof: From Theorem 2 it follows that

$$C(n, r) = \frac{n!}{r!\,(n-r)!}$$

and

$$C(n, n-r) = \frac{n!}{(n-r)!\,[n-(n-r)]!} = \frac{n!}{(n-r)!\,r!}.$$

Hence, $C(n, r) = C(n, n-r)$. □

There is another common notation for the number of r-combinations from a set with n elements, namely,

$$\binom{n}{r}.$$

This number is also called a **binomial coefficient**. The name *binomial coefficient* is used because these numbers occur as coefficients in the expansion of powers of binomial expressions such as $(a+b)^n$. We will discuss the **binomial theorem**, which expresses a power of a binomial expression as a sum of terms involving binomial coefficients, later in this section.

until there are exactly $n - r + 1$ ways to choose the rth element. Consequently, by the product rule, there are

$$n(n-1)(n-2)\cdots(n-r+1)$$

r-permutations of the set. ☐

From Theorem 1 it follows that

$$P(n, r) = n(n-1)(n-2)\cdots(n-r+1) = n!/(n-r)!$$

In particular, note that $P(n, n) = n!$. We will illustrate this result with some examples.

EXAMPLE 2 How many different ways are there to select 4 different players from 10 players on a team to play four tennis matches, where the matches are ordered?

Solution: The answer is given by the number of 4-permutations of a set with 10 elements. By Theorem 1, this is $P(10, 4) = 10 \cdot 9 \cdot 8 \cdot 7 = 5040$. ■

EXAMPLE 3 Suppose that there are eight runners in a race. The winner receives a gold medal, the second-place finisher receives a silver medal, and the third-place finisher receives a bronze medal. How many different ways are there to award these medals, if all possible outcomes of the race can occur?

Solution: The number of different ways to award the medals is the number of 3-permutations of a set with 8 elements. Hence, there are $P(8, 3) = 8 \cdot 7 \cdot 6 = 336$ possible ways to award the medals. ■

EXAMPLE 4 Suppose that a saleswoman has to visit eight different cities. She must begin her trip in a specified city, but she can visit the other seven cities in any order she wishes. How many possible orders can the saleswoman use when visiting these cities?

Solution: The number of possible paths between the cities is the number of permutations of seven elements, since the first city is determined, but the remaining seven can be ordered arbitrarily. Consequently, there are $7! = 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 5040$ ways for the saleswoman to choose her tour. If, for instance, the saleswoman wishes to find the path between the cities with minimum distance, and she computes the total distance for each possible path, she must consider a total of 5040 paths! ■

COMBINATIONS

web

An **r-combination** of elements of a set is an unordered selection of r elements from the set. Thus, an r-combination is simply a subset of the set with r elements.

EXAMPLE 5 Let S be the set $\{1, 2, 3, 4\}$. Then $\{1, 3, 4\}$ is a 3-combination from S. ■

The number of r-combinations of a set with n distinct elements is denoted by $C(n, r)$.

EXAMPLE 6 We see that $C(4, 2) = 6$, since the 2-combinations of $\{a, b, c, d\}$ are the six subsets $\{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}$, and $\{c, d\}$. ■

We can determine the number of r-combinations of a set with n elements using the formula for the number of r-permutations of a set. To do this, note that the r-permutations of a set can be obtained by first forming r-combinations and then ordering the elements in these combinations. The proof of the following theorem, which gives the value of $C(n, r)$, is based on this observation.

THEOREM 2 **The number of r-combinations of a set with n elements, where n is a positive integer and r is an integer with $0 \le r \le n$, equals**

$$C(n, r) = \frac{n!}{r!\,(n-r)!}.$$

Proof: The r-permutations of the set can be obtained by forming the $C(n, r)$ r-combinations of the set, and then ordering the elements in each r-combination, which can be done in $P(r, r)$ ways. Consequently,

$$P(n, r) = C(n, r) \cdot P(r, r).$$

This implies that

$$C(n, r) = \frac{P(n, r)}{P(r, r)} = \frac{n!/(n-r)!}{r!/(r-r)!} = \frac{n!}{r!\,(n-r)!}. \qquad \square$$

The following corollary is helpful in computing the number of r-combinations of a set.

COROLLARY 1 **Let n and r be nonnegative integers with $r \le n$. Then $C(n, r) = C(n, n - r)$.**

Proof: From Theorem 2 it follows that

$$C(n, r) = \frac{n!}{r!\,(n-r)!}$$

and

$$C(n, n-r) = \frac{n!}{(n-r)!\,[n-(n-r)]!} = \frac{n!}{(n-r)!\,r!}.$$

Hence, $C(n, r) = C(n, n - r)$. □

There is another common notation for the number of r-combinations from a set with n elements, namely,

$$\binom{n}{r}.$$

This number is also called a **binomial coefficient.** The name *binomial coefficient* is used because these numbers occur as coefficients in the expansion of powers of binomial expressions such as $(a + b)^n$. We will discuss the **binomial theorem,** which expresses a power of a binomial expression as a sum of terms involving binomial coefficients, later in this section.

EXAMPLE 7 How many ways are there to select 5 players from a 10-member tennis team to make a trip to a match at another school?

Solution: The answer is given by the number of 5-combinations of a set with 10 elements. By Theorem 2, the number of such combinations is

$$C(10,5) = \frac{10!}{5!\,5!} = 252. \qquad \blacksquare$$

EXAMPLE 8 How many ways are there to select a committee to develop a discrete mathematics course at a school if the committee is to consist of 3 faculty members from the mathematics department and 4 from the computer science department, if there are 9 faculty members of the mathematics department and 11 of the computer science department?

Solution: By the product rule, the answer is the product of the number of 3-combinations of a set with 9 elements and the number of 4-combinations of a set with 11 elements. By Theorem 2, the number of ways to select the committee is

$$C(9,3)\cdot C(11,4) = \frac{9!}{3!6!}\cdot\frac{11!}{4!7!} = 84\cdot 330 = 27{,}720. \qquad \blacksquare$$

BINOMIAL COEFFICIENTS

Some of the more important properties of the binomial coefficients will be discussed here. The first property to be discussed is an important identity.

THEOREM 3 PASCAL'S IDENTITY **Let n and k be positive integers with $n \geq k$. Then**

$$C(n+1,k) = C(n,k-1) + C(n,k).$$

Proof: Suppose that T is a set containing $n+1$ elements. Let a be an element in T, and let $S = T - \{a\}$. Note that there are $C(n+1,k)$ subsets of T containing k elements. However, a subset of T with k elements either contains a together with $k-1$ elements of S, or contains k elements of S and does not contain a. Since there are $C(n,k-1)$ subsets of $k-1$ elements of S, there are $C(n,k-1)$ subsets of k elements of T that contain a. And there are $C(n,k)$ subsets of k elements of T that do not contain a, since there are $C(n,k)$ subsets of k elements of S. Consequently,

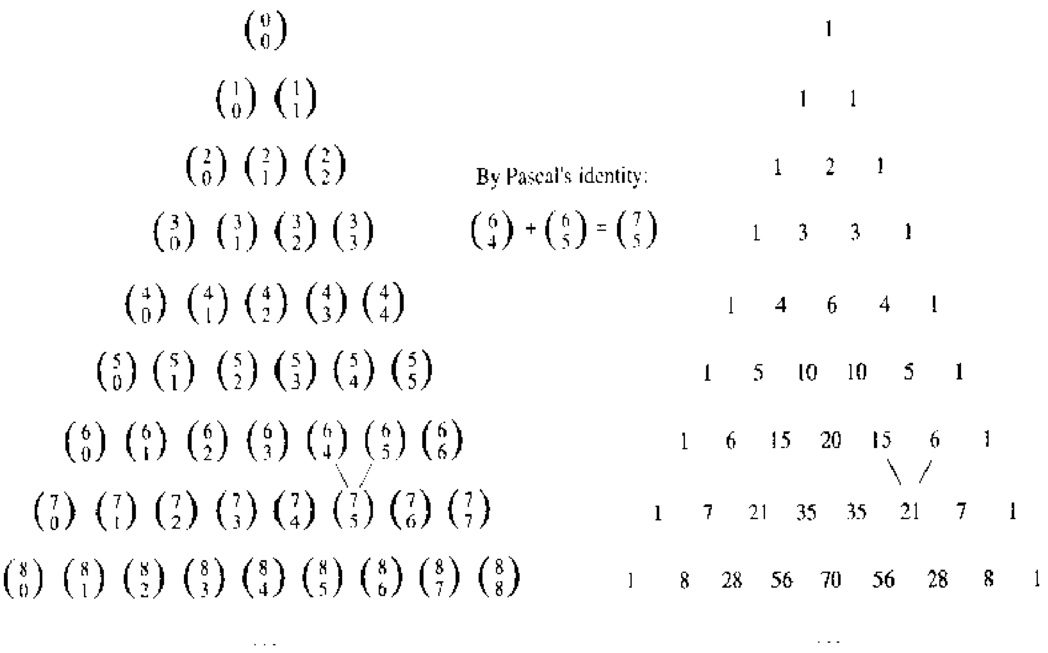
$$C(n+1,k) = C(n,k-1) + C(n,k). \qquad \square$$

Remark: A combinatorial proof of Pascal's identity has been given. It is also possible to prove this identity by algebraic manipulation from the formula for $C(n,r)$ (see Exercise 47 at the end of this section).

Pascal's identity is the basis for a geometric arrangement of the binomial coefficients in a triangle, as shown in Figure 1.

The nth row in the triangle consists of the binomial coefficients

$$\binom{n}{k}, \quad k = 0, 1, \ldots, n.$$



(a) (b)

FIGURE 1 **Pascal's Triangle.**

This triangle is known as **Pascal's triangle.** Pascal's identity shows that when two adjacent binomial coefficients in this triangle are added, the binomial coefficient in the next row between these two coefficients is produced.

The binomial coefficients enjoy many other identities besides Pascal's identity. Two other identities will be stated here. Combinatorial proofs will be given. Others may be found in the exercises at the end of the section.

THEOREM 4 **Let n be a positive integer. Then**

$$\sum_{k=0}^{n} C(n, k) = 2^n.$$

Proof: A set with n elements has a total of 2^n different subsets. Each subset has either zero elements, one element, two elements, . . . , or n elements in it. There are $C(n, 0)$

Blaise Pascal (1623–1662). Blaise Pascal exhibited his talents at an early age, although his father, who had made discoveries in analytic geometry, kept mathematics books away from him to encourage other interests. At 16 Pascal discovered an important result concerning conic sections. At 18 he designed a calculating machine, which he built and sold. Pascal, along with Fermat, laid the foundations for the modern theory of probability. In this work he made new discoveries concerning what is now called Pascal's triangle. In 1654, Pascal abandoned his mathematical pursuits to devote himself to theology. After this, he returned to mathematics only once. One night, distracted by a severe toothache, he sought comfort by studying the mathematical properties of the cycloid. Miraculously, his pain subsided, which he took as a sign of divine approval of the study of mathematics.

subsets with zero elements, $C(n, 1)$ subsets with one element, $C(n, 2)$ subsets with two elements, …, and $C(n, n)$ subsets with n elements. Therefore,

$$\sum_{k=0}^{n} C(n, k)$$

counts the total number of subsets of a set with n elements. This shows that

$$\sum_{k=0}^{n} C(n, k) = 2^n.$$

□

THEOREM 5

VANDERMONDE'S IDENTITY Let m, n, and r be nonnegative integers with r not exceeding either m or n. Then

$$C(m + n, r) = \sum_{k=0}^{r} C(m, r - k)C(n, k).$$

Remark: This identity was discovered by mathematician Alexandre-Théophile Vandermonde in the eighteenth century.

Proof: Suppose that there are m items in one set and n items in a second set. Then the total number of ways to pick r elements from the union of these sets is $C(m + n, r)$. Another way to pick r elements from the union is to pick k elements from the first set and then $r - k$ elements from the second set, where k is an integer with $0 \le k \le r$. This can be done in $C(m, k)C(n, r - k)$ ways, using the product rule. Hence, the total number of ways to pick r elements from the union also equals

$$C(m + n, r) = \sum_{k=0}^{r} C(m, r - k)C(n, k).$$

□

This proves Vandermonde's identity.

THE BINOMIAL THEOREM

The binomial theorem gives the coefficients of the expansion of powers of binomial expressions. A **binomial** expression is simply the sum of two terms, such as $x + y$. (The terms can be products of constants and variables, but that does not concern us here.) The following example illustrates why this theorem holds.

EXAMPLE 9

The expansion of $(x + y)^3$ can be found using combinatorial reasoning instead of multiplying the three terms out. When $(x + y)^3 = (x + y)(x + y)(x + y)$ is expanded, all

---

Alexandre-Théophile Vandermonde (1735–1796). Because Alexandre-Théophile Vandermonde was a sickly child, his physician father directed him to a career in music. However, he later developed an interest in mathematics. His complete mathematical work consists of four papers published in 1771–1772. These papers include fundamental contributions on the roots of equations, on the theory of determinants, and on the knight's tour problem (introduced in the exercises in Section 7.5). Vandermonde's interest in mathematics lasted for only 2 years. Afterward, he published papers on harmony, experiments with cold, and the manufacture of steel. He also became interested in politics, joining the cause of the French revolution and holding several different positions in government.

products of a term in the first sum, a term in the second sum, and a term in the third sum are added. Terms of the form x^3, x^2y, xy^2, and y^3 arise. To obtain a term of the form x^3, an x must be chosen in each of the sums, and this can be done in only one way. Thus, the x^3 term in the product has a coefficient of 1. To obtain a term of the form x^2y, an x must be chosen in two of the three sums (and consequently a y in the other sum). Hence, the number of such terms is the number of 2-combinations of three objects, namely, $C(3, 2)$. Similarly, the number of terms of the form xy^2 is the number of ways to pick one of the three sums to obtain an x (and consequently take a y from each of the other two terms). This can be done in $C(3, 1)$ ways. Finally, the only way to obtain a y^3 term is to choose the y for each of the three sums in the product, and this can be done in exactly one way. Consequently, it follows that

$$(x + y)^3 = x^3 + 3x^2y + 3xy^2 + y^3.$$ ■

The binomial theorem will now be stated.

THEOREM 6 **THE BINOMIAL THEOREM** **Let x and y be variables, and let n be a positive integer. Then**

$$(x+y)^n = \sum_{j=0}^{n} C(n,j)x^{n-j}y^j$$
$$= \binom{n}{0}x^n + \binom{n}{1}x^{n-1}y + \binom{n}{2}x^{n-2}y^2 + \cdots + \binom{n}{n-1}xy^{n-1} + \binom{n}{n}y^n.$$

Proof: A combinatorial proof of the theorem will be given. The terms in the product when it is expanded are of the form $x^{n-j}y^j$ for $j = 0, 1, 2, \ldots, n$. To count the number of terms of the form $x^{n-j}y^j$, note that to obtain such a term it is necessary to choose $n - j$ xs from the n sums (so that the other j terms in the product are ys). Therefore, the coefficient of $x^{n-j}y^j$ is $C(n, n - j) = C(n, j)$. This proves the theorem. ☐

The use of the binomial theorem is illustrated by the following examples.

EXAMPLE 10 What is the expansion of $(x + y)^4$?

Solution: From the binomial theorem it follows that

$$(x+y)^4 = \sum_{j=0}^{4} C(4,j)x^{4-j}y^j$$
$$= C(4,0)x^4 + C(4,1)x^3y + C(4,2)x^2y^2 + C(4,3)xy^3 + C(4,4)y^4$$
$$= x^4 + 4x^3y + 6x^2y^2 + 4xy^3 + y^4.$$ ■

EXAMPLE 11 What is the coefficient of $x^{12}y^{13}$ in the expansion of $(x + y)^{25}$?

Solution: From the binomial theorem it follows that this coefficient is

$$C(25, 13) = \frac{25!}{13!\,12!} = 5{,}200{,}300.$$ ■

EXAMPLE 12 What is the coefficient of $x^{12}y^{13}$ in the expansion of $(2x - 3y)^{25}$?

Solution: First, note that this expression equals $(2x + (-3y))^{25}$. By the binomial theorem, we have

$$(2x + (-3y))^{25} = \sum_{j=0}^{25} C(25, j)(2x)^{25-j}(-3y)^j.$$

Consequently, the coefficient of $x^{12}y^{13}$ in the expansion is obtained when $j = 13$, namely,

$$C(25, 13)2^{12}(-3)^{13} = -\frac{25!}{13!\,12!}2^{12}3^{13}.$$

■

The binomial theorem can be used to give another proof of Theorem 4. Recall that this theorem states that $\sum_{k=0}^{n} C(n, k) = 2^n$ whenever n is a positive integer.

Proof: Using the binomial theorem we see that

$$2^n = (1 + 1)^n = \sum_{k=0}^{n} C(n, k)1^k 1^{n-k} = \sum_{k=0}^{n} C(n, k).$$

This is the desired result. □

The binomial theorem can also be used to prove the following identity.

THEOREM 7 **Let n be a positive integer. Then**

$$\sum_{k=0}^{n} (-1)^k C(n, k) = 0.$$

Proof: From the binomial theorem it follows that

$$0 = ((-1) + 1)^n = \sum_{k=0}^{n} C(n, k)(-1)^k 1^{n-k} = \sum_{k=0}^{n} C(n, k)(-1)^k.$$

This proves the theorem. □

Exercises

1. List all the permutations of $\{a, b, c\}$.

2. How many permutations are there of the set $\{a, b, c, d, e, f, g\}$?

3. How many permutations of $\{a, b, c, d, e, f, g\}$ end with a?

4. Let $S = \{1, 2, 3, 4, 5\}$.
  a) List all the 3-permutations of S.
  b) List all the 3-combinations of S.

5. Find the value of each of the following quantities.
  a) $P(6, 3)$ **b)** $P(6, 5)$ **c)** $P(8, 1)$
  d) $P(8, 5)$ **e)** $P(8, 8)$ **f)** $P(10, 9)$

6. Find the value of each of the following quantities.
  a) $C(5, 1)$ **b)** $C(5, 3)$ **c)** $C(8, 4)$
  d) $C(8, 8)$ **e)** $C(8, 0)$ **f)** $C(12, 6)$

7. Find the number of 5-permutations of a set with nine elements.

8. In how many different orders can five runners finish a race if no ties are allowed?

9. How many possibilities are there for the win, place, and show (first, second, and third) positions in a horse race with 12 horses if all orders of finish are possible?

10. There are six different candidates for governor of a state. In how many different orders can the names of the candidates be printed on a ballot?

11. A group contains n men and n women. How many ways are there to arrange these people in a row if the men and women alternate?

12. In how many ways can a set of two positive integers less than 100 be chosen?

13. In how many ways can a set of five letters be selected from the English alphabet?

14. How many subsets with an odd number of elements does a set with 10 elements have?

15. How many subsets with more than two elements does a set with 100 elements have?

16. How many bit strings of length 10 have
- **a)** exactly three 0s?
- **b)** the same number of 0s as 1s?
- **c)** at least seven 1s?
- **d)** at least three 1s?

17. One hundred tickets, numbered 1, 2, 3, . . . , 100, are sold to 100 different people for a drawing. Four different prizes are awarded, including a grand prize (a trip to Tahiti). How many ways are there to award the prizes if
- **a)** there are no restrictions?
- **b)** the person holding ticket 47 wins the grand prize?
- **c)** the person holding ticket 47 wins one of the prizes?
- **d)** the person holding ticket 47 does not win a prize?
- **e)** the people holding tickets 19 and 47 both win prizes?
- **f)** the people holding tickets 19, 47, and 73 all win prizes?
- **g)** the people holding tickets 19, 47, 73, and 97 all win prizes?
- **h)** none of the people holding tickets 19, 47, 73, and 97 wins a prize?
- **i)** the grand prize winner is a person holding ticket 19, 47, 73, or 97?
- **j)** the people holding tickets 19 and 47 win prizes, but the people holding tickets 73 and 97 do not win prizes?

18. Thirteen people on a softball team show up for a game.
- **a)** How many ways are there to choose 10 players to take the field?
- **b)** How many ways are there to assign the 10 positions by selecting players from the 13 people who show up?
- **c)** Of the 13 people who show up, 3 are women. How many ways are there to choose 10 players to take the field if at least one of these players must be a woman?

19. A club has 25 members.
- **a)** How many ways are there to choose four members of the club to serve on an executive committee?
- **b)** How many ways are there to choose a president, vice president, secretary, and treasurer of the club?

20. A professor writes 40 discrete mathematics true/false questions. Of the statements in these questions, 17 are true. If the questions can be positioned in any order, how many different answer keys are possible?

21. How many 4-permutations of the positive integers not exceeding 100 contain three consecutive integers in the correct order
- **a)** where consecutive means in the usual order of the integers and where these consecutive integers can perhaps be separated by other integers in the permutation?
- **b)** where consecutive means both that the numbers be consecutive integers and that they be in consecutive positions in the permutation?

22. Seven women and nine men are on the faculty in the mathematics department at a school.
- **a)** How many ways are there to select a committee of five members of the department if at least one woman must be on the committee?
- **b)** How many ways are there to select a committee of five members of the department if at least one woman and at least one man must be on the committee?

23. The English alphabet contains 21 consonants and five vowels. How many strings of six lowercase letters of the English alphabet contain
- **a)** exactly 1 vowel?
- **b)** exactly 2 vowels?
- **c)** at least 1 vowel?
- **d)** at least 2 vowels?

24. How many strings of six lowercase letters from the English alphabet contain
- **a)** the letter a?
- **b)** the letters a and b?
- **c)** the letters a and b in consecutive positions with a preceding b, with all the letters distinct?
- **d)** the letters a and b, where a is somewhere to the left of b in the string, with all the letters distinct?

25. Suppose that a department contains 10 men and 15 women. How many ways are there to form a committee with 6 members if it must have the same number of men and women?

26. Suppose that a department contains 10 men and 15 women. How many ways are there to form a committee with 6 members if it must have more women than men?

27. How many bit strings contain exactly eight 0s and ten 1s if every 0 must be immediately followed by a 1?

28. How many bit strings contain exactly five 0s and fourteen 1s if every 0 must be immediately followed by two 1s?

29. How many bit strings of length 10 contain at least three 1s and at least three 0s?

30. How many ways are there to select 12 countries in the United Nations to serve on a council if 3 are selected from a block of 45, 4 are selected from a block of 57, and the others are selected from the remaining 69 countries?

31. How many license plates consisting of three letters followed by three digits contain no letter or digit twice?

32. How many ways are there to seat six people around a circular table, where seatings are considered to be the same if they can be obtained from each other by rotating the table?

33. Show that if n and k are positive integers, then

$$C(n+1,k) = (n+1)C(n,k-1)/k.$$

Use this identity to construct an inductive definition of the binomial coefficients.

34. Show that if p is a prime and k is an integer such that $1 \leq k \leq p-1$, then p divides $C(p,k)$.

35. Find the expansion of $(x+y)^5$.

36. Find the coefficient of x^5y^8 in $(x+y)^{13}$.

37. How many terms are there in the expansion of $(x+y)^{100}$?

38. What is the coefficient of x^7 in $(1+x)^{11}$?

39. What is the coefficient of x^9 in $(2-x)^{19}$?

40. What is the coefficient of x^8y^9 in the expansion of $(3x+2y)^{17}$?

41. What is the coefficient of $x^{101}y^{99}$ in the expansion of $(2x-3y)^{200}$?

***42.** Give a formula for the coefficient of x^k in the expansion of $(x+1/x)^{100}$, where k is an integer.

***43.** Give a formula for the coefficient of x^k in the expansion of $(x^2-1/x)^{100}$, where k is an integer.

44. The row of Pascal's triangle containing the binomial coefficients $C(10,k)$, $0 \leq k \leq 10$, is:

1 10 45 120 210 252 210 120 45 10 1

Use Pascal's identity to produce the row immediately following this row in Pascal's triangle.

45. What is the row of Pascal's triangle containing the binomial coefficients $C(9,k)$, $0 \leq k \leq 9$?

***46.** Let n be a positive integer. What is the largest binomial coefficient $C(n,r)$, where r is a nonnegative integer less than or equal to n? Prove your answer is correct.

47. Prove Pascal's identity, using the formula for $C(n,r)$.

48. Prove the identity $C(n,r)C(r,k) = C(n,k) \cdot C(n-k, r-k)$, whenever n, r, and k are nonnegative integers with $r \leq n$ and $k \leq r$,

a) using a combinatorial argument.

b) using an argument based on the formula for the number of r-combinations of a set with n elements.

***49.** Prove that

$$\sum_{k=0}^{r} C(n+k,k) = C(n+r+1,r),$$

whenever n and r are positive integers,

a) using a combinatorial argument.

b) using Pascal's identity.

50. Show that if n is a positive integer, then $C(2n,2) = 2C(n,2)+n^2$

a) using a combinatorial argument.

b) by algebraic manipulation.

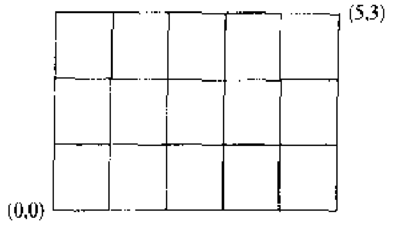
***51.** Give a combinatorial proof that $\sum_{k=1}^{n} kC(n,k) = n2^{n-1}$. (*Hint:* Count in two ways the number of ways to select a committee and to then select a leader of the committee.)

***52.** Give a combinatorial proof that $\sum_{k=1}^{n} kC(n,k)^2 = nC(2n-1,n-1)$. (*Hint:* Count in two ways the number of ways to select a committee, with n members from a group of n mathematics professors and n computer science professors, such that the chairperson of the committee is a mathematics professor.)

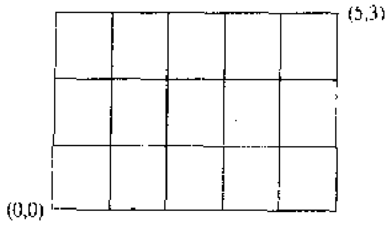
53. Show that a set has the same number of subsets with an odd number of elements as it does subsets with an even number of elements.

***54.** Prove the binomial theorem using mathematical induction.

55. In this exercise we will count the number of paths in the xy plane between the origin (0,0) and point (m,n) such that each path is made up of a series of steps, where each step is a move one unit to the right or a move one unit upward. (No moves to the left or downward are allowed.) Two such paths from (0,0) to (5,3) are illustrated here.



(a)



(b)

a) Show that each path of the type described can be represented by a bit string consisting of m 0s and n

1s, where a 0 represents a move one unit to the right and a 1 represents a move one unit upward.

b) Conclude from part (a) that there are $C(m + n, n)$ paths of the desired type.

56. Use Exercise 55 to prove that $C(n, k) = C(n, n - k)$ whenever k is an integer with $0 \le k \le n$. [*Hint:* Consider the number of paths of the type described in Exercise 55 from $(0, 0)$ to $(n - k, k)$ and from $(0, 0)$ to $(k, n - k)$.]

57. Use Exercise 55 to prove Theorem 4. [*Hint:* Count the number of paths with n steps of the type described in Exercise 55. Every such path must end at one of the points $(n - k, k)$ for $k = 0, 1, 2, \ldots, n$.]

58. Use Exercise 55 to prove Pascal's identity. [*Hint:* Show that a path of the type described in Exercise 55 from $(0, 0)$ to $(n + 1 - k, k)$ passes through either $(n + 1 - k, k - 1)$ or $(n - k, k)$, but not through both.]

59. Prove the identity in Exercise 49 using Exercise 55. [*Hint:* First, note that the number of paths from $(0, 0)$ to $(n + 1, r)$ equals $C(n + 1 + r, r)$. Second, count the number of paths by summing the number of these paths that start by going k units upward for $k = 0, 1, 2, \ldots, r$.]

***60.** How many ways are there for a horse race with four horses to finish if ties are possible? (Note that since ties are allowed, any number of the four horses may tie.)

***61.** There are six runners in the 100-yard dash. How many ways are there for three medals to be awarded if ties are possible? (The runner or runners who finish with the fastest time receive gold medals, the runner or runners who finish with exactly one runner ahead receive silver medals, and the runner or runners who finish with exactly two runners ahead receive bronze medals.)

***62.** The following procedure is used to break ties in games in the championship round of the World Cup soccer tournament. Each team selects five players in a prescribed order. Each of these players takes a penalty kick, with a player from the first team followed by a player from the second team and so on, following the order of players specified. If the score is still tied at the end of the ten penalty kicks, this procedure is repeated. If the score is till tied after 20 penalty kicks, a sudden-death shootout occurs, with the first team scoring an unanswered goal victorious.

a) How many different scoring scenarios are possible if the game is settled in the first round of 10 penalty kicks, where the round ends once it is impossible for a team to equal the number of goals scored by the other team?

b) How many different scoring scenarios for the first and second groups of penalty kicks are possible if the game is settled in the second round of 10 penalty kicks?

c) How many scoring scenarios are possible for the full set of penalty kicks if the game is settled with no more than 10 total additional kicks after the two rounds of 5 kicks for each team?

***63.** Determine a formula involving binomial coefficients for the nth term of a sequence if its initial terms are those listed. (*Hint:* Looking at Pascal's triangle will be helpful. Although infinitely many sequences start with a specified set of terms, each of the following lists is the start of a sequence of the type desired.)

a) 1, 3, 6, 10, 15, 21, 28, 36, 45, 55, 66, ...

b) 1, 4, 10, 20, 35, 56, 84, 120, 165, 220, ...

c) 1, 2, 6, 20, 70, 252, 924, 3432, 12870, 48620, ...

d) 1, 1, 2, 3, 6, 10, 20, 35, 70, 126, ...

e) 1, 1, 1, 3, 1, 5, 15, 35, 1, 9, ...

f) 1, 3, 15, 84, 495, 3003, 18564, 116280, 735471, 4686825, ...

4.4

Discrete Probability

INTRODUCTION

Combinatorics and probability theory share common origins. The theory of probability was first developed in the seventeenth century when certain gambling games were analyzed by the French mathematician Blaise Pascal. It was in these studies that Pascal discovered various properties of the binomial coefficients. In the eighteenth century, the French mathematician Laplace, who also studied gambling, gave a definition of the probability of an event as the number of successful outcomes divided by the number of possible outcomes. For instance, the probability a die comes up an odd number when it is rolled is the number of successful outcomes—namely, the number of ways it can come up odd—divided by the number of possible outcomes—namely, the number of different ways the die can come up. There are a total of six possible outcomes—namely, 1, 2, 3, 4, 5, and 6—and exactly three of these are successful outcomes—namely, 1, 3, and 5. Hence, the probability that the die comes up an odd number is $3/6 = 1/2$. (Note that it has been assumed that all possible outcomes are equally likely, or, in other words, that the die is fair.)

In this section we will restrict ourselves to experiments that have finitely many, equally likely, outcomes. This permits us to use Laplace's definition of the probability of an event. We will continue our study of probability in Section 4.5, where we will study experiments with finitely many outcomes that are not necessarily equally likely. In Section 4.5 we will also introduce some key concepts in probability theory, including conditional probability, independence of events, random variables, and expected values.

FINITE PROBABILITY

An **experiment** is a procedure that yields one of a given set of possible outcomes. The **sample space** of the experiment is the set of possible outcomes. An **event** is a subset of the sample space. Laplace's definition of the probability of an event with finitely many possible outcomes will now be stated.

DEFINITION 1. The *probability* of an event E, which is a subset of a finite sample space S of equally likely outcomes, is $p(E) = |E|/|S|$.

Some additional examples are given here.

EXAMPLE 1 An urn contains four blue balls and five red balls. What is the probability that a ball chosen from the urn is blue?

Solution: To calculate the probability, note that there are nine possible outcomes, and four of these possible outcomes produce a blue ball. Hence, the probability that a blue ball is chosen is $4/9$. ■

EXAMPLE 2 What is the probability that when two dice are rolled, the sum of the numbers on the two dice is 7?

Pierre Simon Laplace (1749–1827). Pierre Simon Laplace came from humble origins in Normandy. In his childhood he was educated in a school run by the Benedictines. At 16 he entered the University of Caen intending to study theology. However, he soon realized his true interests were in mathematics. After completing his studies he was named a provisional professor at Caen, and in 1769 he became professor of mathematics at the Paris Military School.

Laplace is best known for his contributions to celestial mechanics, the study of the motions of heavenly bodies. His *Traité du Mécanique Céleste* is considered one of the greatest scientific works of the early nineteenth century. Laplace was one of the founders of probability theory and made many contributions to mathematical statistics. His work in this area is documented in his book *Thèorie Analytique des Probabilitées*, in which he defined the probability of an event as the ratio of the number of favorable outcomes to the total number of outcomes of an experiment.

Laplace was famous for his political flexibility. He was loyal, in succession, to the French Republic, Napoleon, and King Louis XVIII. This permitted him to be productive before, during, and after the French revolution.

Solution: There are a total of 36 possible outcomes when two dice are rolled. (The product rule can be used to see this; since each die has 6 possible outcomes, the total number of outcomes when two dice are rolled is $6^2 = 36$.) There are 6 successful outcomes, namely, (1, 6), (2, 5), (3, 4), (4, 3), (5, 2), and (6, 1), where the values of the first and second dice are represented by an ordered pair. Hence, the probability that a 7 comes up when two fair dice are rolled is $6/36 = 1/6$. ■

Lotteries have become extremely popular recently. We can easily compute the odds of winning different types of lotteries.

EXAMPLE 3 In a lottery, players win a large prize when they pick four digits that match, in the correct order, four digits selected by a random mechanical process. A smaller prize is won if only three digits are matched. What is the probability that a player wins the large prize? What is the probability that a player wins the small prize?

Solution: There is only one way to choose all four digits correctly. By the product rule, there are $10^4 = 10{,}000$ ways to choose four digits. Hence, the probability that a player wins the large prize is $1/10{,}000 = 0.0001$.

Players win the smaller prize when they correctly choose exactly three of the four digits. Exactly one digit must be wrong to get three digits correct, but not all four correct. By the sum rule, the number of ways to choose exactly three digits correctly can be obtained by adding the number of ways to choose four digits matching the digits picked in all but the ith position, for $i = 1, 2, 3, 4$. To count the number of successes with the first digit incorrect, note that there are nine possible choices for the first digit (all but the one correct digit), and one choice for each of the other digits, namely, the correct digits for these slots. Hence, there are nine ways to choose four digits where the first digit is incorrect, but the last three are correct. Similarly, there are nine ways to choose four digits where the second digit is incorrect, nine with the third digit incorrect, and nine with the fourth digit incorrect. Hence, there is a total of 36 ways to choose four digits with exactly three of the four digits correct. Thus, the probability that a player wins the smaller prize is $36/10{,}000 = 9/2500 = 0.0036$. ■

EXAMPLE 4 There are many lotteries now that award enormous prizes to people who correctly choose a set of six numbers out of the first n positive integers, where n is usually between 30 and 50. What is the probability that a person picks the correct six numbers out of 40?

Solution: There is only one winning combination. The total number of ways to choose six numbers out of 40 is

$$C(40, 6) = \frac{40!}{34!\,6!} = 3{,}838{,}380.$$

Consequently, the probability of picking a winning combination is $1/3{,}838{,}380 \sim 0.00000026$. (Here the symbol ~ means approximately equal to.) ■

We can find the probability of hands in card games using the techniques developed so far. A deck of cards contains 52 cards. There are 13 different kinds of cards, with 4 cards of each kind. These kinds are twos, threes, fours, fives, sixes, sevens, eights, nines, tens, jacks, queens, kings, and aces. There are also four suits, spades, clubs, hearts, and diamonds, each containing 13 cards, with one card of each kind in a suit.

EXAMPLE 5 How many different hands of 5 cards from the deck of 52 are there?

Solution: There are $C(52, 5) = 2{,}598{,}960$ different hands with five cards. ■

EXAMPLE 6 Find the probability that a hand of five cards in poker contains four cards of one kind.

Solution: By the product rule, the number of hands of five cards with four cards of one kind is the product of the number of ways to pick one kind, the number of ways to pick the four of this kind out of the four in the deck of this kind, and the number of ways to pick the fifth card. This is

$$C(13, 1)C(4, 4)C(48, 1).$$

Since there is a total of $C(52, 5)$ different hands of five cards, the probability that a hand contains four cards of one kind is

$$\frac{C(13, 1)C(4, 4)C(48, 1)}{C(52, 5)} = \frac{13 \cdot 1 \cdot 48}{2{,}598{,}960} \sim 0.00024.$$ ■

EXAMPLE 7 What is the probability that a poker hand contains a full house, that is, three of one kind and two of another kind?

Solution: By the product rule, the number of hands containing a full house is the product of the number of ways to pick two kinds in order, the number of ways to pick three out of four for the first kind, and the number of ways to pick two out of four for the second kind. (Note that the order of the two kinds matters, since, for instance, three queens and two aces is different than three aces and two queens.) We see that the number of hands containing a full house is

$$P(13, 2)C(4, 3)C(4, 2) = 13 \cdot 12 \cdot 4 \cdot 6 = 3744.$$

Since there are 2,598,960 poker hands, the probability of a full house is

$$\frac{3744}{2{,}598{,}960} \sim 0.0014.$$ ■

THE PROBABILITY OF COMBINATIONS OF EVENTS

We can use counting techniques to find the probability of events derived from other events.

THEOREM 1 **Let E be an event in a sample space S. The probability of the event $\overline{E}$, the complementary event of E, is given by**

$$p(\overline{E}) = 1 - p(E).$$

Proof: To find the probability of the event $\overline{E}$, note that $|\overline{E}| = |S| - |E|$. Hence,

$$p(\overline{E}) = \frac{|S| - |E|}{|S|} = 1 - \frac{|E|}{|S|} = 1 - p(E).$$ □

There is an alternative strategy for finding the probability of an event when a direct approach does not work well. Instead of determining the probability of the event, the probability of its complement can be found. This is often easier to do, as the following example shows.

EXAMPLE 8 A sequence of 10 bits is randomly generated. What is the probability that at least one of these bits is 0?

Solution: Let E be the event that at least one of the 10 bits is 0. Then $\overline{E}$ is the event that all the bits are 1s. Since the sample space S is the set of all bit strings of length 10, it follows that

$$\begin{aligned} p(E) &= 1 - p(\overline{E}) \\ &= 1 - \frac{|\overline{E}|}{|S|} \\ &= 1 - \frac{1}{2^{10}} \\ &= 1 - \frac{1}{1024} \\ &= \frac{1023}{1024}. \end{aligned}$$

Hence, the probability that the bit string will contain at least one 0 bit is 1023/1024. It is quite difficult to find this probability directly without using Theorem 1. ■

We can also find the probability of the union of two events.

THEOREM 2 **Let E_1 and E_2 be events in the sample space S. Then**

$$p(E_1 \cup E_2) = p(E_1) + p(E_2) - p(E_1 \cap E_2).$$

Proof: Using the formula given in Section 1.4 for the number of elements in the union of two sets, it follows that

$$|E_1 \cup E_2| = |E_1| + |E_2| - |E_1 \cap E_2|.$$

Hence,

$$\begin{aligned} p(E_1 \cup E_2) &= \frac{|E_1 \cup E_2|}{|S|} \\ &= \frac{|E_1| + |E_2| - |E_1 \cap E_2|}{|S|} \\ &= \frac{|E_1|}{|S|} + \frac{|E_2|}{|S|} - \frac{|E_1 \cap E_2|}{|S|} \\ &= p(E_1) + p(E_2) - p(E_1 \cap E_2). \end{aligned}$$ □

EXAMPLE 9 What is the probability that a positive integer selected at random from the set of positive integers not exceeding 100 is divisible by either 2 or 5?

Solution: Let E_1 be the event that the integer selected is divisible by 2, and let E_2 be the event that it is divisible by 5. Then $E_1 \cup E_2$ is the event that it is divisible by either 2 or 5. Also, $E_1 \cap E_2$ is the event that it is divisible by both 2 and 5, or equivalently, that it is divisible by 10. Since $|E_1| = 50$, $|E_2| = 20$, and $|E_1 \cap E_2| = 10$, it follows that

$$\begin{aligned} p(E_1 \cup E_2) &= p(E_1) + p(E_2) - p(E_1 \cap E_2) \\ &= \frac{50}{100} + \frac{20}{100} - \frac{10}{100} \\ &= \frac{3}{5}. \end{aligned}$$

■

PROBABILISTIC REASONING

A common problem is to determine which of two events is more likely. Analyzing the probabilities of such events can be tricky. The following example describes a problem of this type. It discusses a famous problem originating with the television game show "Let's Make a Deal."

EXAMPLE 10 The Monty Hall Three Door Puzzle. Suppose you are a game show contestant. You have a chance to win a large prize. You are asked to select one of three doors to open; the larger prize is behind one of the three doors. Once you select a door, the game show host, who knows what is behind each door, does the following. First, whether or not you selected the winning door, he opens one of the other two doors that he knows is a losing door (selecting at random if both are losing doors). Then he asks you whether you would like to switch doors. Which strategy should you use? Should you change doors or keep your original selection, or does it not matter?

Solution: The probability you select the correct door (before the host opens a door and asks you whether you want to change) is 1/3, since the three doors are equally likely to be the correct door. The probability this is the correct door does not change once the game show host opens one of the other doors, since he will always open a door that the prize is not behind.

The probability that you selected incorrectly is the probability the prize is behind one of the two doors you did not select. Consequently, the probability that you selected incorrectly is 2/3. If you selected incorrectly, when the game show host opens a door to show you that the prize is not behind it, the prize is behind the other door. You will always win if your initial choice was incorrect and you change doors. So, by changing doors, the probability you win is 2/3. In other words, you should always change doors when given the chance to do so by the game show host. This doubles the probability that you will win. ■

Exercises

1. What is the probability that a card selected from a deck is an ace?
2. What is the probability that a die comes up six when it is rolled?

3. What is the probability that a randomly selected integer chosen from the first 100 positive integers is odd?
4. What is the probability that a randomly selected day of the year (from the 366 possible days) is in April?
5. What is the probability that the sum of the numbers on two dice is even when they are rolled?
6. What is the probability that a card selected from a deck is an ace or a heart?
7. What is the probability that a coin lands heads up six times in a row?
8. What is the probability that a five-card poker hand contains the ace of hearts?
9. What is the probability that a five-card poker hand does not contain the queen of hearts?
10. What is the probability that a five-card poker hand contains the two of diamonds and the three of spades?
11. What is the probability that a five-card poker hand contains the two of diamonds, the three of spades, the six of hearts, the ten of clubs, and the king of hearts?
12. What is the probability that a five-card poker hand contains exactly one ace?
13. What is the probability that a five-card poker hand contains at least one ace?
14. What is the probability that a five-card poker hand contains cards of five different kinds?
15. What is the probability that a five-card poker hand contains two pairs (that is, two of each of two different kinds and a fifth card of a third kind)?
16. What is the probability that a five-card poker hand contains a flush, that is, five cards of the same suit?
17. What is the probability that a five-card poker hand contains a straight, that is, five cards that have consecutive kinds? (Note that an ace can be considered either the lowest card of an A-2-3-4-5 straight or the highest card of a 10-J-Q-K-A straight.)
18. What is the probability that a five-card poker hand contains a straight flush, that is, five cards of the same suit of consecutive kinds?

*19. What is the probability that a five-card poker hand contains cards of five different kinds and does not contain a flush or a straight?

20. What is the probability that a five-card poker hand contains a royal flush, that is, the 10, jack, queen, king, and ace of one suit?
21. What is the probability that a die never comes up an even number when it is rolled six times?
22. What is the probability that a positive integer not exceeding 100 selected at random is divisible by 3?
23. What is the probability that a positive integer not exceeding 100 selected at random is divisible by 5 or 7?
24. Find the probability of winning the lottery by selecting the correct six integers, where the order in which these integers are selected does not matter, from the positive integers not exceeding
    a) 30. b) 36. c) 42. d) 48.
25. Find the probability of winning the lottery by selecting the correct six integers, where the order in which these integers are selected does not matter, from the positive integers not exceeding
    a) 50. b) 52. c) 56. d) 60.
26. Find the probability of selecting none of the correct six integers, where the order in which these integers are selected does not matter, from the positive integers not exceeding
    a) 40. b) 48. c) 56. d) 64.
27. Find the probability of selecting exactly one of the correct six integers, where the order in which these integers are selected does not matter, from the positive integers not exceeding
    a) 40. b) 48. c) 56. d) 64.
28. To play the Pennsylvania superlottery, a player selects 7 numbers out of the first 80 positive integers. What is the probability that a person wins the grand prize by picking 7 numbers that are among the 11 numbers selected by the Pennsylvania lottery commission?
29. In a superlottery, players win a fortune if they choose the eight numbers selected by a computer from the positive integers not exceeding 100. What is the probability that a player wins this superlottery?
30. What is the probability that a player wins the prize offered for correctly choosing five (but not six) numbers out of six integers chosen between 1 and 40, inclusive, by a computer?
31. In roulette, a wheel with 38 numbers is spun. Of these, 18 are red, and 18 are black. The other two numbers, which are neither black nor red, are 0 and 00. The probability that when the wheel is spun it lands on any particular number is 1/38.
    a) What is the probability that the wheel lands on a red number?
    b) What is the probability that the wheel lands on a black number twice in a row?
    c) What is the probability that the wheel lands on 0 or 00?
    d) What is the probability that the wheel does not land on 0 or 00 five times in a row?
    e) What is the probability that the wheel lands on a number between 1 and 6, inclusive, on one spin, but does not land between them on the next spin?
32. Which is more likely, rolling a total of 8 when two dice are rolled or rolling a total of 8 when three dice are rolled?
33. Which is more likely, rolling a total of 9 when two dice are rolled or rolling a total of 9 when three dice are rolled?
34. Two events E_1 and E_2 are called **independent** if $p(E_1 \cap E_2) = p(E_1)p(E_2)$. For each of the following pairs of events, which are subsets of the set of all possible outcomes when a coin is tossed three times, determine whether or not they are independent.

a) E_1: the first coin comes up tails; E_2: the second coin comes up heads.
b) E_1: the first coin comes up tails; E_2: two, and not three, heads come up in a row.
c) E_1: the second coin comes up tails; E_2: two, and not three, heads come up in a row.

(We will study independence of events in more depth in Section 4.5.)

35. Explain what is wrong with the statement that in the Monty Hall Three Door Puzzle the probability that the prize is behind the first door you select and the probability that the prize is behind the other of the two doors that Monty does not open are both 1/2, since there are two doors left.

36. Suppose that instead of three doors, there are four doors in the Monty Hall puzzle. What is the probability that you win by not changing once the host, who knows what is behind each door, opens a losing door and gives you the chance to change doors? What is the probability that you win by changing the door you select to one of the two remaining doors among the three that you did not select?

4.5

Probability Theory

INTRODUCTION

In Section 4.4 we introduced the notion of the probability of an event. (Recall that an event is a subset of the possible outcomes of an experiment.) We defined the probability of an event E as Laplace did, that is,

$$p(E) = \frac{|E|}{|S|},$$

the number of outcomes in E divided by the total number of outcomes. This definition assumes that all outcomes are equally likely. However, many experiments have outcomes that are not equally likely. For instance, a coin may be biased so that it comes up heads twice as often as tails. Similarly, the likelihood that the input of a linear search is a particular element in a list, or is not in the list, depends on how the input is generated. How can we model the likelihood of events in such situations? In this section we will show how to define probabilities of outcomes to study probabilities of experiments where outcomes may not be equally likely.

Suppose that a fair coin is flipped four times, and the first time it comes up heads. Given this information, what is the probability that heads comes up three times? To answer this and similar questions, we will introduce the concept of *conditional probability*. Does knowing that the first flip comes up heads change the probability that heads comes up three times? If not, these two events are called *independent*, a concept studied later in this section.

Many questions address a particular numerical value associated with the outcome of an experiment. For instance, when we flip a coin 100 times, what is the probability that exactly 40 heads appear? How many heads should we expect to appear? In this section we will study *random variables*, which are functions that associate numerical values to the outcomes of experiments, and their weighted averages, called *expected values*.

ASSIGNING PROBABILITIES

Let S be the sample space of an experiment with a finite or countable number of outcomes. We assign a probability $p(s)$ to each outcome s. We require that two conditions be met:

$(i)\quad 0 \le p(s) \le 1$ for each $s \in S$

and

$$(ii)\quad \sum_{s \in S} p(s) = 1.$$

Condition (i) states that the probability of each outcome is a nonnegative real number no greater than 1. Condition (ii) states that the sum of the probabilities of all possible outcomes should be 1; that is, when we do the experiment, it is a certainty that one of these outcomes occurs. This is a generalization of Laplace's definition in which each of n outcomes is assigned a probability of $1/n$. Indeed, conditions (i) and (ii) are met when Laplace's definition of probabilities of equally likely outcomes is used. (See Exercise 4.)

Note when there are n possible outcomes, $x_1, x_2, \ldots, x_n$, the two conditions to be met are

$(i)\quad 0 \le p(x_i) \le 1$ for $i = 1, 2, \ldots, n$

and

$$(ii)\quad \sum_{i=1}^{n} p(x_i) = 1.$$

To model an experiment, the probability $p(s)$ assigned to an outcome s should equal the limit of the number of times s occurs divided by the number of times the experiment is performed, as this number grows without bound. (We will assume that all experiments discussed have outcomes that are predictable on the average, so that this limit exists. We also assume that the outcomes of successive trials of an experiment do not depend on past results.)

Remark: In this section we will require that the number of possible outcomes is finite. A countably infinite number of outcomes can be handled similarly using infinite series, as illustrated in Exercises 37–40 at the end of this section. We will not discuss probabilities of events when the set of outcomes is not discrete, such as when the outcome of an experiment can be any real number. In such cases integral calculus is usually required for the study of the probabilities of events.

We can model experiments in which outcomes are either equally likely or not equally likely by choosing the appropriate function $p(s)$, as Example 1 illustrates.

EXAMPLE 1 What probabilities should we assign to the outcomes H (heads) and T (tails) when a fair coin is flipped? What probabilities should be assigned to these events when the coin is biased so that heads comes up twice as often as tails?

Solution: For a fair coin, the probability that heads comes up when the coin is flipped equals the probability that tails comes up, so that the events are equally likely. Consequently, we assign the probability 1/2 to each of the two possible outcomes, that is, $p(H) = p(T) = 1/2$.

For the biased coin we have

$$p(H) = 2p(T).$$

Since

$$p(H) + p(T) = 1,$$

it follows that

$$2p(T) + p(T) = 3p(T) = 1.$$

We conclude that $p(T) = 1/3$ and $p(H) = 2/3$. ■

We now define the probability of an event as the sum of the probabilities of the outcomes in this event.

DEFINITION 1. The *probability* of the event E is the sum of the probabilities of the outcomes in E. That is,

$$p(E) = \sum_{s \in E} p(s).$$

Note that when there are n outcomes in the event E, that is, if $E = \{a_1, a_2, \ldots, a_n\}$, then $p(E) = \sum_{i=1}^{n} p(a_i)$.

EXAMPLE 2 Suppose that a die is biased (or loaded) so that 3 appears twice as often as each other number but that the other five outcomes are equally likely. What is the probability that an odd number appears when we roll this die?

Solution: We want to find the probability of the event $E = \{1, 3, 5\}$. By Exercise 2 at the end of this section, we have

$$p(1) = p(2) = p(4) = p(5) = p(6) = 1/7; p(3) = 2/7.$$

It follows that

$$p(E) = p(1) + p(3) + p(5) = 1/7 + 2/7 + 1/7 = 4/7.$$ ■

When events are equally likely and there are a finite number of possible outcomes, the definition of the probability of an event given in this section (Definition 1) agrees with Laplace's definition (Definition 1 of Section 4.4). To see this, suppose that there are n equally likely outcomes; each possible outcome has probability $1/n$, since the sum of their probabilities is 1. Suppose the event E contains m outcomes. According to Definition 1,

$$p(E) = \sum_{i=1}^{m} \frac{1}{n} = \frac{m}{n}.$$

Since $|E| = m$ and $|S| = n$, it follows that

$$p(E) = \frac{m}{n} = \frac{|E|}{|S|}.$$

This is Laplace's definition of the probability of the event E.

COMBINATIONS OF EVENTS

The formulae for probabilities of combinations of events in Section 4.4 continue to hold when we use Definition 1 to define the probability of an event. For example, Theorem 1 of Section 4.4 asserts that

$$p(\bar{E}) = 1 - p(E)$$

where $\bar{E}$ is the complementary event of the event E. This equality also holds when Definition 1 is used. To see this, note that since the sum of the probabilities of the n possible outcomes is 1, and each outcome is either in E or in $\bar{E}$, but not in both, we have

$$\sum_{s \in S} p(s) = 1 = p(E) + p(\bar{E}).$$

Hence, $p(\bar{E}) = 1 - p(E)$.

Under Laplace's definition, by Theorem 2 in Section 4.4, we have

$$p(E_1 \cup E_2) = p(E_1) + p(E_2) - p(E_1 \cap E_2)$$

whenever E_1 and E_2 are events in a sample space S. This also holds when we define the probability of an event as we do in this section. To see this, note that $p(E_1 \cup E_2)$ is the sum of the probabilities of the outcomes in $E_1 \cup E_2$. When an outcome x is in one, but not both, of E_1 and E_2, $p(x)$ occurs in exactly one of the sums for $p(E_1)$ and $p(E_2)$. When an outcome x is in both E_1 and E_2, $p(x)$ occurs in the sum for $p(E_1)$, in the sum for $p(E_2)$, and in the sum for $p(E_1 \cap E_2)$, so that it occurs $1 + 1 - 1 = 1$ time on the right-hand side. Consequently, the left-hand side and right-hand side are equal.

CONDITIONAL PROBABILITY

Suppose that we flip a coin three times, and all eight possibilities are equally likely. Moreover, suppose we know that the event F, that the first flip comes up tails, occurs. Given this information, what is the probability of the event E, that an odd number of tails appears? Since the first flip comes up tails, there are only four possible outcomes: TTT, TTH, THT, and THH, where H and T represent heads and tails, respectively. An odd number of tails appears only for the outcomes TTT and THH. Since the eight outcomes have equal probability, each of the four possible outcomes, given that F occurs, should also have an equal probability of 1/4. This suggests that we should assign the probability of $2/4 = 1/2$ to E, given that F occurs. This probability is called the **conditional probability** of E given F.

In general, to find the conditional probability of E given F, we use F as the sample space. For an outcome from E to occur, this outcome must also belong to $E \cap F$. With this motivation, we make the following definition.

DEFINITION 2. Let E and F be events with $p(F) > 0$. The *conditional probability* of E given F, denoted by $p(E \mid F)$, is defined as

$$p(E \mid F) = \frac{p(E \cap F)}{p(F)}.$$

EXAMPLE 3 What is the probability that a bit string of length four, generated at random so that each of the 16 bit strings of length four is equally likely, contains at least two consecutive 0s, given that its first bit is a 0? (We assume that 0 bits and 1 bits are equally likely.)

Solution: Let E be the event that a bit string of length four contains at least two consecutive 0s, and let F be the event that the first bit of a bit string of length four is a 0. The probability that a bit string of length four has at least two consecutive 0s, given that its first bit is a 0, equals

$$p(E \mid F) = \frac{p(E \cap F)}{p(F)}.$$

Since $E \cap F = \{0000, 0001, 0010, 0011, 0100\}$, we see that $p(E \cap F) = 5/16$. Since there are eight bit strings of length four that start with a 0, we have $p(F) = 8/16 = 1/2$. Consequently,

$$p(E \mid F) = \frac{5/16}{1/2} = \frac{5}{8}.$$

■

EXAMPLE 4 What is the conditional probability that a family with two children has two boys, given they have at least one boy? Assume that each of the possibilities BB, BG, GB, and GG is equally likely, where B represents a boy and G represents a girl.

Solution: Let E be the event that a family with two children has two boys, and let F be the event that a family with two children has at least one boy. It follows that $E = \{BB\}$, $F = \{BB, BG, GB\}$, and $E \cap F = \{BB\}$. Since the four possibilities are equally likely, it follows that $p(F) = 3/4$ and $p(E \cap F) = 1/4$. We conclude that

$$p(E \mid F) = \frac{p(E \cap F)}{p(F)} = \frac{1/4}{3/4} = \frac{1}{3}.$$

■

INDEPENDENCE

Suppose a coin is flipped four times, as described in the introduction to our discussion of conditional probability. Does knowing that the first flip comes up tails (event F), alter the probability that tails comes up an odd number of times (event E)? In other words, is it the case that $p(E \mid F) = p(E)$? This equality is valid for the events E and F, since $p(E \mid F) = 1/2$ and $p(E) = 1/2$. Because this equality holds, we say that E and F are **independent events.**

Since $p(E \mid F) = p(E \cap F)/p(F)$, asking whether $p(E \mid F) = p(E)$ is the same as asking whether $p(E \cap F) = p(E)p(F)$. This leads to the following definition.

DEFINITION 3. **The events E and F are said to be *independent* if and only if $p(E \cap F) = p(E)p(F)$.**

EXAMPLE 5 Suppose E is the event that a randomly generated bit string of length four begins with a 1 and F is the event that a randomly generated bit string contains an even number of 0s. Are E and F independent, if the 16 bit strings of length four are equally likely?

Solution: There are eight bit strings of length four that begin with a 1: 1000, 1001, 1010, 1011, 1100, 1101, 1110, and 1111. There are also eight bit strings of length four that contain an even number of 1s: 0000, 0011, 0101, 0110, 1001, 1010, 1100, 1111. Since there are 16 bit strings of length four, it follows that

$$p(E) = p(F) = 8/16 = 1/2.$$

Because $E \cap F = \{1111, 1100, 1010, 1001\}$, we see that

$$p(E \cap F) = 4/16 = 1/4.$$

Since

$$p(E \cap F) = 1/4 = (1/2)(1/2) = p(E)p(F),$$

we conclude that E and F are independent. ■

EXAMPLE 6 Assume, as in Example 4, that each of the four ways a family can have two children is equally likely. Are the events E, that a family with two children has two boys, and F, that a family with two children has at least one boy, independent?

Solution: Since $E = \{BB\}$, we have $p(E) = 1/4$. In Example 4 we showed that $p(F) = 3/4$ and that $p(E \cap F) = 1/4$. Since $p(E \cap F) = 1/4 \neq 3/16 = (1/4)(3/4) = p(E)p(F)$, events E and F are not independent. ■

EXAMPLE 7 Are the events E, that a family with three children has children of both sexes, and F, that a family with three children has at most one boy, independent? Assume that the eight ways a family can have three children are equally likely.

Solution: By assumption, each of the eight ways a family can have three children, BBB, BBG, BGB, BGG, GBB, GBG, GGB, and GGG, has a probability of 1/8. Since $E = \{BBG, BGB, BGG, GBB, GBG, GGB\}$, $F = \{BGG, GBG, GGB, GGG\}$, and $E \cap F = \{BGG, GBG, GGB\}$, it follows that $p(E) = 6/8 = 3/4$, $p(F) = 4/8 = 1/2$, and $p(E \cap F) = 3/8$. Since

$$p(E \cap F) = \frac{3}{8} = \frac{3}{4} \cdot \frac{1}{2} = p(E)p(F),$$

we conclude that E and F are independent. (This conclusion may seem surprising. Indeed, if we change the number of children, the conclusion may no longer hold. See Exercise 19 at the end of this section.) ■

BERNOULLI TRIALS AND THE BINOMIAL DISTRIBUTION

Suppose that an experiment can have only two possible outcomes. For instance, when a bit is generated at random, the possible outcomes are 0 and 1. When a coin is flipped, the possible outcomes are heads and tails. Each performance of an experiment with two possible outcomes is called a **Bernoulli trial,** after James Bernoulli, who made important contributions to probability theory. In general, a possible outcome of a Bernoulli trial is called a **success** or a **failure.** If p is the probability of a success and q is the probability of a failure, it follows that $p + q = 1$.

Many problems can be solved by determining the probability of k successes when an experiment consists of n independent Bernoulli trials. Consider the following example.

EXAMPLE 8 A coin is biased so that the probability of heads is 2/3. What is the probability that exactly four heads come up when the coin is flipped seven times, assuming that the flips are independent?

Solution: There are $2^7 = 128$ possible outcomes when a coin is flipped seven times. The number of ways four of the seven flips can be heads is $C(7, 4)$. Since the seven flips are independent, the probability of each of these outcomes is $(2/3)^4(1/3)^3$. Consequently, the probability that exactly four heads appear is

$$\begin{aligned} C(7,4)(2/3)^4(1/3)^3 &= \frac{35 \cdot 16}{3^7} \\ &= \frac{560}{2187}. \end{aligned}$$

■

Following the same reasoning as used in Example 8, we can establish the following theorem, which tells us the probability of k successes in n independent Bernoulli trials.

THEOREM 1 PROBABILITY OF k SUCCESSES IN n INDEPENDENT BERNOULLI TRIALS **The probability of k successes in n independent Bernoulli trials, with probability of success p and probability of failure $q = 1 - p$, is**

$$C(n, k)p^k q^{n-k}.$$

Proof: When n Bernoulli trials are carried out, the outcome is an n-tuple $(t_1, t_2, \ldots, t_n)$, where $t_i = S$ (for success) or $t_i = F$ (for failure) for $i = 1, 2, \ldots, n$. Since the n trials are independent, the probability of each outcome of n trials consisting of k successes and $n - k$ failures (in any order) is $p^k q^{n-k}$. Since there are $C(n, k)$ n-tuples of Ss and Fs that contain k Ss, the probability of k successes is

$$C(n, k)p^k q^{n-k}.$$

□

---

web

James Bernoulli (1654–1705). James Bernoulli (also known as Jacob I), was born in Basel, Switzerland. He is one of the eight prominent mathematicians in the Bernoulli family (see Section 8.1 for the Bernoulli family tree of mathematicians). Following his father's wish, James studied theology and entered the ministry. But contrary to the desires of his parents, he also studied mathematics and astronomy. He traveled throughout Europe from 1676 to 1682, learning about the latest discoveries in mathematics and the sciences. Upon returning to Basel in 1682, he founded a school for mathematics and the sciences. He was appointed professor of mathematics at the University of Basel in 1687, remaining in this position for the rest of his life.

James Bernoulli is best known for the work *Ars Conectandi*, published 8 years after his death. In this work, he described the known results in probability theory and in enumeration, often providing alternative proofs of known results. This work also includes the application of probability theory to games of chance and his introduction of the theorem known as the **law of large numbers**. This law states that if $\epsilon > 0$, as n becomes arbitrarily large the probability approaches 1 that the number of times an event E occurs during n trials is within ϵ of $p(E)$.

We denote by $b(k; n, p)$ the probability of k successes in n independent Bernoulli trials with probability of success p and probability of failure $q = 1 - p$. Considered as a function of k, we call this function the **binomial distribution.** Theorem 1 tells us that $b(k; n, p) = C(n, k)p^k q^{n-k}$.

EXAMPLE 9 What is the probability that exactly eight 0 bits are generated when 10 bits are generated with the probability that a 0 bit is generated is 0.9, the probability that a 1 bit is generated is 0.1, and the bits are generated independently?

Solution: By Theorem 1, the probability that exactly eight 0 bits are generated is

$$b(8; 10, 0.9) = C(10, 8)(0.9)^8(0.1)^2 = 0.1937102445. \qquad \blacksquare$$

Note that the sum of the probabilities that there are k successes when n independent Bernoulli trials are carried out, for $k = 0, 1, 2, \ldots, n$, equals

$$\sum_{k=0}^{n} C(n, k)p^k q^{n-k} = (p + q)^n = 1,$$

as should be the case. The first equality in this string of equalities is a consequence of the binomial theorem. The second equality follows since $q = 1 - p$.

RANDOM VARIABLES

Many problems are concerned with a numerical value associated with the outcome of an experiment. For instance, we may want to know the probability that there are nine 1 bits generated when 10 bits are randomly generated, or we may want to know the probability that a coin comes up tails 11 times when it is flipped 20 times. To study problems of this type we introduce the concept of a random variable.

DEFINITION 4. A random variable is a function from the sample space of an experiment to the set of real numbers. That is, a random variable assigns a real number to each possible outcome.

Remark: Note that a random variable is a function. It is not a variable, and it is not random!

EXAMPLE 10 Suppose that a coin is flipped three times. Let $X(t)$ be the number of heads that appear when t is the outcome. Then the random variable $X(t)$ takes on the following values:

$$X(HHH) = 3,$$

$$X(HHT) = X(HTH) = X(THH) = 2,$$

$$X(TTH) = X(THT) = X(HTT) = 1,$$

$$X(TTT) = 0. \qquad \blacksquare$$

EXAMPLE 11 Let X be the sum of the numbers that appear when a pair of dice is rolled. What are the values of this random variable for the 36 possible outcomes (i, j), where i and j are the numbers that appear on the first die and the second die, respectively, when these two dice are rolled?

Solution: The random variable X takes on the following values:

$$\begin{aligned}
&X((1,1)) = 2,\\
&X((1,2)) = X((2,1)) = 3,\\
&X((1,3)) = X((2,2)) = X((3,1)) = 4,\\
&X((1,4)) = X((2,3)) = X((3,2)) = X((4,1)) = 5,\\
&X((1,5)) = X((2,4)) = X((3,3)) = X((4,2)) = X((5,1)) = 6,\\
&X((1,6)) = X((2,5)) = X((3,4)) = X((4,3)) = X((5,2)) = X((6,1)) = 7,\\
&X((2,6)) = X((3,5)) = X((4,4)) = X((5,3)) = X((6,2)) = 8,\\
&X((3,6)) = X((4,5)) = X((5,4)) = X((6,3)) = 9,\\
&X((4,6)) = X((5,5)) = X((6,4)) = 10,\\
&X((5,6)) = X((6,5)) = 11,\\
&X((6,6)) = 12.
\end{aligned}$$

■

EXPECTED VALUES

Many questions can be formulated in terms of the value we expect a random variable to take, or more precisely, the average value of a random variable when an experiment is performed a large number of times. Questions of this kind include: How many heads are expected to appear when a coin is flipped 100 times? What is the expected number of comparisons used to find an element in a list using a linear search? To study such questions we introduce the concept of the expected value of a random variable.

web

DEFINITION 5. **The *expected value* (or *expectation*) of the random variable $X(s)$ on the sample space S is equal to**

$$E(X) = \sum_{s \in S} p(s)X(s).$$

Note when the sample space S has n elements $S = \{x_1, x_2, \ldots, x_n\}$, $E(x) = \sum_{i=1}^{n} p(x_i)X(x_i)$.

Remark: We are concerned only with random variables with finite expected values here.

EXAMPLE 12 A fair coin is flipped three times. Let S be the sample space of the eight possible outcomes, and let X be the random variable that assigns to an outcome the number of heads in this outcome. What is the expected value of X?

Solution: In Example 10 we listed the values of X for the eight possible outcomes when a coin is flipped three times. Since the coin is fair and the flips are independent, the probability of each outcome is 1/8. Consequently,

$$\begin{aligned}E(X) &= \frac{1}{8}(X(HHH) + X(HHT) + X(HTH) + X(THH) + X(TTH)\\ &\quad + X(THT) + X(HTT) + X(TTT))\\ &= \frac{1}{8}(3 + 2 + 2 + 2 + 1 + 1 + 1 + 0)\\ &= \frac{12}{8}\\ &= \frac{3}{2}.\end{aligned}$$

■

When an experiment has relatively few outcomes, we can compute the expected value of a random variable directly from its definition, as was done in Example 12. However, when an experiment has a large number of outcomes, it may be inconvenient to compute the expected value of a random variable directly from its definition. Instead, we can find the expected value of a random variable by grouping together all outcomes assigned the same value by the random variable. In particular, suppose that X is a random variable with range $X(S)$, and let $p(X = r)$ be the probability that the random variable X takes the value r. Consequently, $p(X = r)$ is the sum of the probabilities of the outcomes s such that $X(s) = r$. It follows that

$$E(X) = \sum_{r \in X(S)} p(X = r)r.$$

Examples 13 and 14 illustrate the use of this formula. In Example 13 we will find the expected value of the sum of the numbers that appear on two fair dice when they are rolled. In Example 14 we will find the expected value of the number of successes when n Bernoulli trials are performed.

EXAMPLE 13 What is the expected value of the sum of the numbers that appear when a pair of fair dice is rolled?

Solution: Let X be the random variable equal to the sum of the numbers that appear when a pair of dice is rolled. In Example 11 we listed the value of X for the 36 outcomes of this experiment. The range of X is $\{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$. By Example 11 we see that

$$\begin{aligned}&p(X = 2) = p(X = 12) = 1/36,\\ &p(X = 3) = p(X = 11) = 2/36 = 1/18,\\ &p(X = 4) = p(X = 10) = 3/36 = 1/12,\\ &p(X = 5) = p(X = 9) = 4/36 = 1/9,\\ &p(X = 6) = p(X = 8) = 5/36,\\ &p(X = 7) = 6/36 = 1/6.\end{aligned}$$

Substituting these values in the formula, we have

$$\begin{aligned} E(X) &= 2\cdot\frac{1}{36}+3\cdot\frac{1}{18}+4\cdot\frac{1}{12}+5\cdot\frac{1}{9}+6\cdot\frac{5}{36}+7\cdot\frac{1}{6} \\ &\quad +8\cdot\frac{5}{36}+9\cdot\frac{1}{9}+10\cdot\frac{1}{12}+11\cdot\frac{1}{18}+12\cdot\frac{1}{36} \\ &= 7. \end{aligned}$$

■

EXAMPLE 14 What is the expected value of the number of successes when n Bernoulli trials are performed, where p is the probability of success on each trial?

Solution: Let X be the random variable equal to the number of successes in n trials. By Theorem 1 we see that $p(X = k) = C(n, k)p^kq^{n-k}$. Hence, from the formula for the expected value of a random variable that groups together outcomes assigned the same value by the random variable, we have

$$\begin{aligned} E(X) &= \sum_{k=1}^{n} kp(X = k) \\ &= \sum_{k=1}^{n} kC(n, k)p^kq^{n-k} \\ &= \sum_{k=1}^{n} nC(n-1, k-1)p^kq^{n-k} \\ &= np\sum_{k=1}^{n} C(n-1, k-1)p^{k-1}q^{n-k} \\ &= np\sum_{j=0}^{n-1} C(n-1, j)p^jq^{n-1-j} \\ &= np(p+q)^{n-1} \\ &= np. \end{aligned}$$

The third equality follows from the fact that $C(n, k) = nC(n-1, k-1)/k$, which follows from Exercise 33 of Section 4.3. The fifth equality is obtained by shifting the index of summation with $j = k-1$ so that j ranges from 0 to $n-1$ when k ranges from 1 to n. The sixth equality follows from the Binomial Theorem. The seventh equality follows since $p+q = 1$. From this computation, we conclude that the expected value of X equals np. This means that the expected number of successes in n Bernoulli trials is np. ■

Theorem 2 establishes some useful properties of expected values, including that the expected value of the sum of random variables is the sum of their expected values.

THEOREM 2 **If X and Y are random variables on a space S, then $E(X + Y) = E(X) + E(Y)$. Furthermore, if X_i, $i = 1, 2, \ldots, n$ with n a positive integer, are random variables on S, then $E(X) = E(X_1) + E(X_2) + \cdots + E(X_n)$. Moreover, if a and b are real numbers, then $E(aX + b) = aE(X) + b$.**

Proof: The first result follows directly from the definition of expected value, since

$$\begin{aligned} E(X+Y) &= \sum_{s\in S} p(s)(X(s)+Y(s)) \\ &= \sum_{s\in S} p(s)X(s) + \sum_{s\in S} p(s)Y(s) \\ &= E(X)+E(Y). \end{aligned}$$

The case with n random variables follows easily using mathematical induction from the case of two random variables. Finally, note that $E(aX+b) = \sum_{s\in S} p(s)(aX(s)+b) = a\sum_{s\in S} p(s)X(s) + b\sum_{s\in S} p(s) = aE(X)+b$ since $\sum_{s\in S} p(s) = 1$. □

Theorem 2 can be useful for computing expected values, since many random variables are sums of simpler random variables, as Examples 15 and 16 illustrate.

EXAMPLE 15 Use Theorem 2 to find the expected value of the sum of the numbers that appear when a pair of fair dice is rolled. (This was done in Example 13 without the benefit of this theorem.)

Solution: Let X_1 and X_2 be the random variables with $X_1((i,j)) = i$ and $X_2((i,j)) = j$, so that X_1 is the number appearing on the first die and X_2 is the number appearing on the second die. It is easy to see that $E(X_1) = E(X_2) = 7/2$ since both equal $(1+2+3+4+5+6)/6 = 21/6 = 7/2$. The sum of the two numbers that appear when the two dice are rolled is the sum X_1+X_2. By Theorem 2, the expected value of the sum is $E(X_1+X_2) = E(X_1)+E(X_2) = 7/2+7/2 = 7$. ∎

EXAMPLE 16 In Example 14, the expected value of the number of successes when n Bernoulli trials are performed, where p is the probability of success on each trial, was shown by direct computation to be np. Show how this result can be found using Theorem 2.

Solution: Let X_i be the random variable with $X_i((t_1,t_2,\ldots,t_n)) = 1$ if t_i is a success and $X_i((t_1,t_2,\ldots,t_n)) = 0$ if t_i is a failure. The expected value of X_i is $E(X_i) = 1\cdot p + 0\cdot(1-p) = p$ for $i = 1,2,\ldots,n$. Let $X = X_1+X_2+\cdots+X_n$ so that X counts the number of successes when these n Bernoulli trials are performed. Theorem 2, applied to the sum of n random variables, shows that $E(X) = E(X_1)+E(X_2)+\cdots+E(X_n) = np$. ∎

We have already discussed independent events. We will now define what it means for two random variables to be independent.

INDEPENDENT RANDOM VARIABLES

DEFINITION 6 **The random variables X and Y on a sample space S are *independent* if**

$$p(X(s) = r_1 \text{ and } Y(s) = r_2) = p(X(s) = r_1)\cdot p(Y(s) = r_2),$$

or in words, if the probability that $X(s) = r_1$ and $Y(s) = r_2$ equals the product of the probabilities that $X(s) = r_1$ and $Y(s) = r_2$, for all real numbers r_1 and r_2.

EXAMPLE 17 Are the random variables X_1 and X_2 from Example 15 independent?

Solution: Let $S = \{1, 2, 3, 4, 5, 6\}$, and let $i \in S$ and $j \in S$. Since there are 36 possible outcomes when the pair of dice is rolled and each is equally likely, we have

$$p(X_1 = i \text{ and } X_2 = j) = 1/36.$$

Furthermore, $p(X_1 = i) = 1/6$ and $p(X_2 = j) = 1/6$, since the probability that i appears on the first die and the probability that j appears on the second die are both 1/6. It follows that

$$p(X_1 = i \text{ and } X_2 = j) = 1/36 = (1/6)(1/6) = p(X_1 = i)p(X_2 = j),$$

so X_1 and X_2 are independent. ■

EXAMPLE 18 Show that the random variables X_1 and $X = X_1 + X_2$, where X_1 and X_2 are as defined in Example 15, are not independent.

Solution: Note that $p(X_1 = 1 \text{ and } X = 12) = 0$, since $X_1 = 1$ means the number appearing on the first die is 1, which implies that the sum of the numbers appearing on the two dice cannot equal 12. On the other hand, $p(X_1 = 1) = 1/6$ and $p(X = 12) = 1/36$. Hence $p(X_1 = 1 \text{ and } X = 12) \neq p(X_1 = 1) \cdot p(X = 12)$. This counterexample shows that X_1 and X are not independent. ■

The expected value of the product of two independent random variables is the product of their expected values, as Theorem 3 shows.

THEOREM 3 **If X and Y are independent random variables on a space S, then $E(XY) = E(X)E(Y)$.**

Proof: From the definition of expected value and since X and Y are independent random variables, it follows that

$$\begin{aligned} E(XY) &= \sum_{s \in S} X(s)Y(s)p(s) \\ &= \sum_{r_1 \in X(S), r_2 \in Y(S)} r_1 r_2 \cdot p(X(s) = r_1 \text{ and } Y(s) = r_2) \\ &= \sum_{r_1 \in X(S), r_2 \in Y(S)} r_1 r_2 \cdot p(X(s) = r_1) \cdot p(Y(s) = r_2) \\ &= \left(\sum_{r_1 \in X(S)} r_1 p(X(s) = r_1) \right) \cdot \left(\sum_{r_2 \in Y(S)} r_2 p(Y(s) = r_2) \right) \\ &= E(X)E(Y). \end{aligned}$$

This completes the proof. □

VARIANCE

web The expected value of a random variable tells us its average value but nothing about how widely its values are distributed. For example, if X and Y are the random variables on the set $S = \{1, 2, 3, 4, 5, 6\}$, with $X(s) = 0$ for all $s \in S$ and $Y(s) = -1$ if $s \in \{1, 2, 3\}$ and $Y(s) = 1$ if $s \in \{4, 5, 6\}$, then the expected values of X and Y are both zero. However, the random variable X never varies from 0, while the random variable Y always differs from 0 by 1. The variance of a random variable helps us characterize how widely a random variable is distributed.

DEFINITION 7 **Let X be a random variable on a sample space S. The *variance* of X, denoted by $V(X)$, is**

$$V(X) = \sum_{s \in S} (X(s) - E(X))^2 p(s).$$

The *standard deviation* of X, denoted $\sigma(X)$, is defined to be $\sqrt{V(X)}$.

The following theorem provides a useful simple expression for the variance of a random variable.

THEOREM 4 **If X is a random variable on a sample space S, then $V(X) = E(X^2) - E(X)^2$.**

Proof: Note that

$$\begin{aligned} V(X) &= \sum_{s \in S} (X(s) - E(X))^2 p(s) \\ &= \sum_{s \in S} X(s)^2 p(s) - 2E(X) \sum_{s \in S} X(s) p(s) + E(X)^2 \sum_{s \in S} p(s) \\ &= E(X^2) - 2E(X)E(X) + E(X)^2 \\ &= E(X^2) - E(X)^2 \end{aligned}$$

We have used the fact that $\sum_{s \in S} p(s) = 1$ in the next-to-last step. □

EXAMPLE 19 What is the variance of the random variable X with $X(t) = 1$ if a Bernoulli trial is a success and $X(t) = 0$ if it is a failure, where p is the probability of success?

Solution: Since X takes only the values 0 and 1, it follows that $X^2(t) = X(t)$. Hence,

$$V(X) = E(X^2) - E(X)^2 = p - p^2 = p(1 - p) = pq.$$ ■

EXAMPLE 20 What is the variance of the random variable $X((i, j)) = 2i$, where i is the number appearing on the first die and j is the number appearing on the second die, when two dice are rolled?

Solution: We will use Theorem 4 to find the variance of X. To do so, we need to find the expected values of X and X^2. Note that since $p(X = k)$ is 1/6 for $k = 2, 4, 6, 8, 10, 12$ and is 0 otherwise,

$$E(X) = (2 + 4 + 6 + 8 + 10 + 12)/6 = 7,$$

and

$$E(X^2) = (2^2 + 4^2 + 6^2 + 8^2 + 10^2 + 12^2)/6 = 182/3.$$

It follows from Theorem 4 that

$$V(X) = E(X^2) - E(X)^2 = 182/3 - 49 = 35/3. \quad \blacksquare$$

Another useful fact about variances is that the variance of the sum of two independent random variables is the sum of their variances. This result is useful for computing the variance of the result of n independent Bernoulli trials, for instance.

THEOREM 5 **If X and Y are two independent random variables on a sample space S, then $V(X + Y) = V(X) + V(Y)$. Furthermore, if X_i, $i = 1, 2, \ldots, n$, with n a positive integer, are pairwise independent random variables on S, then $V(X_1 + X_2 + \cdots + X_n) = V(X_1) + V(X_2) + \cdots + V(X_n)$.**

Proof: From Theorem 4, we have

$$V(X + Y) = E((X + Y)^2) - E(X + Y)^2.$$

It follows that

$$\begin{aligned} V(X + Y) &= E(X^2 + 2XY + Y^2) - (E(X) + E(Y))^2 \\ &= E(X^2) + 2E(XY) + E(Y^2) - E(X)^2 - 2E(X)E(Y) - E(Y)^2. \end{aligned}$$

Since X and Y are independent, by Theorem 3 we have $E(XY) = E(X)E(Y)$. It follows that

$$\begin{aligned} V(X + Y) &= (E(X^2) - E(X)^2) + (E(Y^2) - E(Y)^2) \\ &= V(X) + V(Y). \end{aligned}$$

The case with n pairwise independent random variables can be proved using mathematical induction; the proof is left to the reader. □

EXAMPLE 21 Find the variance and standard deviation of the random variable X whose value when two dice are rolled is $X((i, j)) = i + j$, where i is the number appearing on the first die and j is the number appearing on the second die.

Solution: Let X_1 and X_2 be the random variables defined by $X_1((i, j)) = i$ and $X_2((i, j)) = j$ for a roll of the dice. Then $X = X_1 + X_2$ and X_1 and X_2 are independent, as Example 17 showed. From Theorem 5 it follows that $V(X) = V(X_1) + V(X_2)$. A simple computation as in Example 20 together with Exercise 43 in the Supplementary Exercise at the end of the chapter tell us that $V(X_1) = V(X_2) = 35/12$. Hence, $V(X) = 35/12 + 35/12 = 35/6$ and $\sigma(X) = \sqrt{35/6}$. ■

We will now find the variance of the random variable that counts the number of successes when n independent Bernoulli trials are carried out.

EXAMPLE 22 What is the variance of the number of successes when n independent Bernoulli trials are performed, where p is the probability of success on each trial?

Solution: Let X_i be the random variable with $X_i((t_1, t_2, \ldots, t_n)) = 1$ if t_i is a success and $X_i((t_1, t_2, \ldots, t_n)) = 0$ if t_i is a failure. Let $X = X_1 + X_2 + \cdots + X_n$. Then X counts the number of successes in the n trials. From Theorem 5 it follows that $V(X) = V(X_1) + V(X_2) + \cdots + V(X_n)$. Using Example 19 we have that $V(X_i) = pq$ for $i = 1, 2, \ldots, n$. It follows that $V(X) = npq$. ■

CHEBYSHEV'S INEQUALITY

How likely is it that a random variable takes a value far from its expected value? The following theorem, called Chebyshev's inequality, helps answer this question by providing an upper bound on the probability that the value of a random variable differs from the expected value of the random variable by more than a specified amount.

THEOREM 6 **Chebyshev's Inequality** **Let X be a random variable on a sample space S with probability function p. If r is a positive real number, then**

$$p(|X(s) - E(X)| \geq r) \leq V(X)/r^2.$$

Proof: Let A be the event

$$A = \{s \in S \mid |X(s) - E(X)| \geq r\}.$$

What we want to prove is that $p(A) \leq V(X)/r^2$. Note that

$$\begin{aligned} V(X) &= \sum_{s \in S} (X(s) - E(X))^2 p(s) \\ &= \sum_{s \in A} (X(s) - E(X))^2 p(s) + \sum_{s \notin A} (X(s) - E(X))^2. \end{aligned}$$

The second sum in this expression is nonnegative, since each of its summands is nonnegative. Also, since for each element s in A, $(X(s) - E(X))^2 \geq r^2$, the first sum in this expression is at least $\sum_{s \in A} r^2 p(s)$. Hence, $V(x) \geq \sum_{s \in A} r^2 p(s) = r^2 p(A)$. This is what we needed to prove. □

Chebyshev's inequality, although applicable to any random variable, often fails to provide a practical estimate for the probability that the value of random variable exceeds its mean by a large amount. This is illustrated by the following example.

Pafnuty Lvovich Chebyshev (1821–1894). Chebyshev was born into the gentry in Okatovo, Russia. His father was a retired army officer who fought against Napoleon. In 1832 the family, with its nine children, moved to Moscow, where Pafnuty completed his high school education at home. He entered the Department of Physics and Mathematics at Moscow University. As a student, he developed a new method for approximating the roots of equations. He graduated from Moscow University in 1841 with a degree in mathematics, and he continued his studies, passing his master's exam in 1843 and completing his master's thesis in 1846.

Chebyshev was appointed in 1847 to a position as an assistant at the University of St. Petersburg. He wrote and defended a thesis in 1847. He became a professor at St. Petersburg in 1860, a position he held until 1882. His book on the theory of congruences written in 1849 was influential in the development of number theory. His work on the distribution of prime numbers was seminal. He proved Bertrand's conjecture that for every integer $n > 3$, there is a prime between n and $2n - 2$. Chebyshev helped develop ideas that were later used to prove the prime number theorem. Chebyshev's work on the approximation of functions using polynomials is used extensively when computers are used to find values of functions. Chebyshev was also interested in mechanics. He studied the conversion of rotary motion into rectilinear motion by mechanical coupling. The Chebyshev parallel motion is three linked bars approximating rectilinear motion.

EXAMPLE 23 Let X be the random variable whose value is the number appearing when a fair die is rolled. We have $E(X) = 7/2$ (see Example 15) and $V(X) = 35/12$ (see Example 20). Since the only possible values of X are 1, 2, 3, 4, 5, and 6, X cannot take a value more than 5/2 from its mean, $E(X) = 7/2$. Hence, $p(|X - 7/2| \geq r) = 0$ if $r > 5/2$. By Chebyshev's inequality we know that $p(|X - 7/2| \geq r) \leq (35/12)/r^2$. For example, when $r = 3$, Chebyshev's inequality tells us that $p(|X - 7/2| \geq 3) \leq (35/12)/9 = 35/108$, which is a poor estimate, since $p(|X - 7/2| \geq 3) = 0$. ■

AVERAGE-CASE COMPUTATIONAL COMPLEXITY

Computing the average-case computational complexity of an algorithm can be interpreted as computing the expected value of a random variable. Let the sample space of an experiment be the set of possible inputs a_j, $j = 1, 2, \ldots, n$, and let the random variable X assign to a_j the number of operations used by the algorithm when given a_j as input. Based on our knowledge of the input, we assign a probability $p(a_j)$ to each possible input value a_j. Then, the average-case complexity of the algorithm is

$$E(X) = \sum_{j=1}^{n} p(a_j)X(a_j).$$

This is the expected value of X.

In Example 24 we will illustrate how to find the average-case computational complexity of the linear search algorithm under different assumptions concerning the probability that the element for which we search is an element of the list.

EXAMPLE 24 Average-Case Computational Complexity of the Linear Search Algorithm We are given an element x and a list of n distinct real numbers. The linear search algorithm, described in Section 2.1, locates x by successively comparing it to each element in the list, terminating when x is located or when all the elements have been examined and it has been determined that x is not in the list. What is the average-case computational complexity of the linear search algorithm if the probability that x is in the list is p and it is equally likely that x is any of the n elements in the list? (There are $n + 1$ possible types of input: the n numbers in the list and a number not in the list, which we treat as a single input.)

Solution: In Example 4 of Section 2.2 we showed that $2i + 1$ comparisons are used if x equals the ith element of the list and, in Example 2 of Section 2.2, that $2n + 2$ comparisons are used if x is not in the list. The probability that x equals a_i, the ith element in the list, is p/n, and the probability that x is not in the list is $q = 1 - p$. It follows that the average-case computational complexity of the linear search algorithm is

$$\begin{aligned} E &= 3p/n + 5p/n + \cdots + (2n + 1)p/n + (2n + 2)q \\ &= \frac{p}{n}(3 + 5 + \cdots + (2n + 1)) + (2n + 2)q \\ &= \frac{p}{n}((n + 1)^2 - 1) + (2n + 2)q \\ &= p(n + 2) + (2n + 2)q. \end{aligned}$$

(The third equality follows from Example 2 of Section 3.2.) For instance, when x is guaranteed to be in the list, we have $p = 1$ (so that the probability that $x = a_i$ is $1/n$ for each i) and $q = 0$. Then $E = n + 2$, as we showed in Example 4 in Section 2.2.

When p, the probability that x is in the list, is $1/2$, it follows that $q = 1 - p = 1/2$, so that $E = (n+2)/2 + n + 1 = (3n+4)/2$. Similarly, if the probability that x is in the list is $3/4$, we have $p = 3/4$ and $q = 1/4$, so that $E = 3(n+2)/4 + (n+1)/2 = (5n+8)/4$.

Finally, when x is guaranteed not to be in the list, we have $p = 0$ and $q = 1$. It follows that $E = 2n + 2$, which is not surprising since we have to search the entire list. ■

Exercises

1. What probability should be assigned to the outcome of heads when a biased coin is tossed, if heads is three times as likely to come up as tails? What probability should be assigned to the outcome of tails?

2. Find the probability of each outcome when a loaded die is rolled, if a 3 is twice as likely to appear as each of the other five numbers on the die.

3. Find the probability of each outcome when a biased die is rolled, if rolling a 2 or rolling a 4 is three times as likely as rolling each of the other four numbers on the die and it is equally likely to roll a 2 or a 4.

4. Show that conditions (i) and (ii) are met under Laplace's definition of probability, when outcomes are equally likely.

5. A pair of dice is loaded. The probability that a 4 appears on the first die is 2/7, and the probability that a 3 appears on the second die is 2/7. Other outcomes for each die appear with probability 1/7. What is the probability of 7 appearing as the sum of the numbers when the two dice are rolled?

6. Suppose that E and F are events such that $p(E) = 0.8$ and $p(F) = 0.6$. Show that $p(E \cap F) \geq 0.4$.

7. Show that if E and F are events, then $p(E \cap F) \geq p(E) + p(F) - 1$. This is known as **Bonferroni's inequality.**

8. Use mathematical induction to prove the following generalization of Bonferroni's inequality:

$$p(E_1 \cap E_2 \cap \cdots \cap E_n) \geq p(E_1) + p(E_2) + \cdots + p(E_n) - (n-1),$$

where $E_1, E_2, \ldots, E_n$ are n events.

9. Show that if $E_1, E_2, \ldots, E_n$ are events from a finite sample space, then

$$p(E_1 \cup E_2 \cup \cdots \cup E_n) \leq p(E_1) + p(E_2) + \cdots + p(E_n).$$

This is known as **Boole's inequality.**

10. Show that if E and F are independent events, then $\overline{E}$ and $\overline{F}$ are also independent events.

11. If E and F are independent events, prove or disprove that $\overline{E}$ and F are necessarily independent events.

Exercises 12–14 involve the probability that at least two people in a group have the same birthday.

12. What is the probability that two people have the same birthday? For this problem assume that all the 366 dates are equally likely as birthdays.

***13.** a) What is the probability that in a group of n people, there are at least two with the same birthday? For this problem assume that all of the 366 dates are equally likely as birthdays. (*Hint:* Find the probability that in a group of n people, the birthdays of all the people are different.)
b) How many people are needed to make the probability greater than 1/2 that at least two people have the same birthday?

***14.** February 29 occurs only in leap years. Years divisible by 4, but not by 100, are always leap years. Years divisible by 100, but not by 400, are not leap years, but years divisible by 400 are leap years.
a) What probability distribution for birthdays should be used to reflect how often February 29 occurs?
b) Answer the question asked in part (a) of Exercise 13 using this probability distribution.

15. What is the conditional probability that exactly four heads appear when a fair coin is flipped five times, given that the first flip came up heads?

16. What is the conditional probability that exactly four heads appear when a fair coin is flipped five times, given that the first flip came up tails?

17. What is the conditional probability that a randomly generated bit string of length four contains at least two consecutive 0s, given that the first bit is a 1? (Assume the probabilities of a 0 and a 1 are the same.)

18. Let E be the event that a randomly generated bit string of length three contains an odd number of 1s, and let F be the event that the string starts with 1. Are E and F independent?

19. Let E and F be the events that a family of n children has children of both sexes and has at most one boy, respectively. Are E and F independent if

a) $n = 2$? **b)** $n = 4$?

c) $n = 5$?

20. Assume that the probability a child is a boy is 0.51 and that the sexes of children born into a family are independent. What is the probability that a family of five children has

a) exactly three boys?

b) at least one boy?

c) at least one girl?

d) all children of the same sex?

21. A group of six people play the game of "odd person out" to determine who will buy refreshments. Each person flips a fair coin. If there is a person whose outcome is not the same as that of any other member of the group, this person has to buy the refreshments. What is the probability that there is an odd person out after the coins are flipped once?

22. Find the probability that a randomly generated bit string of length 10 does not contain a 0 if bits are independent and if

a) a 0 bit and a 1 bit are equally likely.

b) the probability that a bit is a 1 is 0.6.

c) the probability that the ith bit is a 1 is $1/2^i$ for $i = 1, 2, 3, \ldots, 10$.

23. Find the probability that a family with five children does not have a boy, if the sexes of children are independent and if

a) a boy and a girl are equally likely.

b) the probability of a boy is 0.51.

c) the probability that the ith child is a boy is $0.51 - (i/100)$.

24. Find the probability that a randomly generated bit string of length 10 begins with a 1 or ends with a 00 for the same conditions as in parts (a), (b), and (c) of Exercise 22, if bits are generated independently.

25. Find the probability that the first child of a family with five children is a boy or that the last two children of the family are girls, for the same conditions as in parts (a), (b), and (c) of Exercise 23.

26. Find each of the following probabilities when n independent Bernoulli trials are carried out with probability of success p.

a) the probability of no successes

b) the probability of at least one success

c) the probability of at most one success

d) the probability of at least two successes

27. Find each of the following probabilities when n independent Bernoulli trials are carried out with probability of success p.

a) the probability of no failures

b) the probability of at least one failure

c) the probability of at most one failure

d) the probability of at least two failures

28. What is the expected number of heads that come up when a fair coin is flipped 10 times?

29. What is the expected number of times a 6 appears when a fair die is rolled 10 times?

30. A coin is biased so that the probability a head comes up when it is flipped is 0.6. What is the expected number of heads that come up when it is flipped 10 times?

31. What is the expected sum of the numbers that appear on two dice, each biased so that a 3 comes up twice as often as each other number?

32. What is the expected value when a \$1 lottery ticket is bought in which the purchaser wins exactly \$10 million if the ticket contains the six winning numbers chosen from the set $\{1, 2, 3, \ldots, 50\}$ and the purchaser wins nothing otherwise?

33. The final exam of a discrete mathematics course consists of 50 true/false questions, each worth two points, and 25 multiple-choice questions, each worth four points. The probability that Linda answers a true/false question correctly is 0.9, and the probability that she answers a multiple-choice question correctly is 0.8. What is her expected score on the final?

34. What is the expected sum of the numbers that appear when three fair dice are rolled?

35. Suppose that the probability that x is in a list of n distinct integers is 2/3 and that it is equally likely that x equals any element in the list. Find the average number of comparisons used by the linear search algorithm to find x or to determine that it is not in the list.

***36.** Suppose the probability that x is the ith element in a list of n distinct integers is $i/[n(n + 1)]$. Find the average number of comparisons used by the linear search algorithm to find x or to determine that it is not in the list.

In this section we have studied experiments with finitely many outcomes. In Exercises 37–40 we study an experiment with countably many outcomes. A coin is flipped until it comes up tails. The sample space of the experiment is

$$\{T, HT, HHT, HHHT, HHHHT, \ldots\}.$$

The probability the coin comes up tails is p.

37. What is the probability that the experiment ends after n flips, that is, the outcome consists of $n - 1$ heads and a tail?

38. Show that the sum of the probabilities of the possible outcomes equals 1.

39. What is the probability that at most n flips are required for the experiment to end?

40. What is the expected value of the number of flips required for the experiment to end?

41. At a party n people toss their hats into a pile in a closet. The hats are mixed up, and each person selects one at random. What is the expected number of people who select their own hats?

42. Let $X(s)$ be a random variable, where $X(s)$ is a nonnegative integer for all $s \in S$, and let A_k be the event that $X(s) \geq k$. Show that $E(X) = \sum_{k=1}^{\infty} p(A_k)$.

43. What is the variance of the number of heads that come up when a fair coin is flipped 10 times?

44. What is the variance of the number of times a 6 appears when a fair die is rolled 10 times?

45. Let X_n be the random variable that counts the difference in the number of tails and the number of heads when n coins are flipped.

a) What is the expected value of X_n?

b) What is the variance of X_n?

46. Provide an example that shows that the variance of the sum of two random variables is not necessarily equal to the sum of their variances when the random variables are not independent.

47. Let X be a random variable on a sample space S such that $X(s) \geq 0$ for all $s \in S$. Show that $p(X(s) \geq a) \leq E(X)/a$ for every positive real number a. This inequality is called **Markov's inequality**.

48. Suppose that the number of cans of soda pop filled in a day at a bottling plant is a random variable with an expected value of 10,000 and a variance of 1000.

a) Use Markov's inequality (Exercise 47) to obtain an upper bound on the probability that the plant will fill more than 11,000 cans on a particular day.

b) Use Chebyshev's inequality to obtain a lower bound on the probability that the plant will fill between 9000 and 11,000 cans on a particular day.

49. Suppose that the number of tin cans recycled in a day at a recycling center is a random variable with an expected value of 50,000 and a variance of 2500.

a) Use Markov's inequality (Exercise 47) to find an upper bound on the probability that the center will recycle more than 55,000 cans on a particular day.

b) Use Chebyshev's inequality to provide a lower bound on the probability that the center will recycle 40,000 to 60,000 cans on a certain day.

The **covariance** of two random variables X and Y on a sample space S, denoted by $\mathrm{Cov}(X, Y)$, is defined to be the expected value of the random variable $(X - E(X))(Y - E(Y))$. That is, $\mathrm{Cov}(X, Y) = E((X - E(X))(Y - E(Y)))$.

50. Show that $\mathrm{Cov}(X, Y) = E(XY) - E(X)E(Y)$, and use this result to conclude that $\mathrm{Cov}(X, Y) = 0$ if X and Y are independent random variables.

51. Show that $V(X + Y) = V(X) + V(Y) + 2\,\mathrm{Cov}(X, Y)$.

52. Find $\mathrm{Cov}(X, Y)$ if X and Y are the random variables with $X((i, j)) = 2i$ and $Y((i, j)) = i + j$, where i and j are the numbers that appear on the first and second of two dice when two fair dice are rolled.

4.6

Generalized Permutations and Combinations

INTRODUCTION

In many counting problems, elements may be used repeatedly. For instance, a letter or digit may be used more than once on a license plate. When a dozen donuts are selected, each variety can be chosen repeatedly. This contrasts with the counting problems discussed earlier in the chapter where we only considered permutations and combinations in which each item could be used at most once. In this section we will show how to solve counting problems where elements may be used more than once.

Also, some counting problems involve indistinguishable elements. For instance, to count the number of ways the letters of the word *SUCCESS* can be rearranged, the placement of identical letters must be considered. This contrasts with the counting problems discussed earlier where all elements were considered distinguishable. In this section we will describe how to solve counting problems in which some elements are indistinguishable.

Moreover, in this section we will explain how to solve another important class of counting problem, problems involving counting the ways to place distinguishable elements in boxes. An example of this type of problem is the number of different ways poker hands can be dealt to four players.

Taken together, the methods described earlier in this chapter and the methods introduced in this section form a useful toolbox for solving a wide range of counting problems. When the additional methods discussed in Chapter 5 are added to this arsenal, you will be able to solve a large percentage of the counting problems that arise in a wide range of areas of study.

PERMUTATIONS WITH REPETITION

Consider the following example of a counting problem when repetition is allowed.

EXAMPLE 1 How many strings of length n can be formed from the English alphabet?

Solution: By the product rule, since there are 26 letters, and since each letter can be used repeatedly, we see that there are 26^n strings of length n. ■

The following question involving probability also involves permutations with repetition.

EXAMPLE 2 What is the probability of drawing three red balls in a row from an urn containing five red balls and seven blue balls if a ball is put back into the urn after it is drawn?

Solution: By the product rule the number of successful outcomes—that is, the number of ways to draw three red balls—is 5^3, since for each drawing there are five red balls in the urn. The total number of outcomes is 12^3, since for each drawing there are 12 balls in the urn. Thus, the desired probability is $5^3/12^3 = 125/1728$. This is an example of **sampling with replacement.** ■

The number of r-permutations of a set with n elements when repetition is allowed is given in the following theorem.

THEOREM 1 **The number of r-permutations of a set of n objects with repetition allowed is n^r.**

Proof: There are n ways to select an element of the set for each of the r positions in the r-permutation when repetition is allowed, since for each choice all n objects are available. Hence, by the product rule there are n^r r-permutations when repetition is allowed. □

COMBINATIONS WITH REPETITION

Consider the following examples of combinations with repetition of elements allowed.

EXAMPLE 3 How many ways are there to select four pieces of fruit from a bowl containing apples, oranges, and pears if the order in which the pieces are selected does not matter, only the type of fruit and not the individual piece matters, and there are at least four pieces of each type of fruit in the bowl?

Solution: To solve this problem we list all the ways possible to select the fruit. There are 15 ways:

4 apples
3 apples, 1 orange
3 oranges, 1 pear
2 apples, 2 oranges
2 apples, 1 orange, 1 pear
4 oranges
3 apples, 1 pear
3 pears, 1 apple
2 apples, 2 pears
2 oranges, 1 apple, 1 pear
4 pears
3 oranges, 1 apple
3 pears, 1 orange
2 oranges, 2 pears
2 pears, 1 apple, 1 orange

The solution is the number of 4-combinations with repetition allowed from a three-element set, {*apple, orange, pear*}. ∎

To solve more complex counting problems of this type, we need a general method for counting the r-combinations of an n-element set. In Example 4 we will illustrate such a method.

EXAMPLE 4 How many ways are there to select five bills from a money bag containing $1 bills, $2 bills, $5 bills, $10 bills, $20 bills, $50 bills, and $100 bills? Assume that the order in which the bills are chosen does not matter, that the bills of each denomination are indistinguishable, and that there are at least five bills of each type.

Solution: Since the order in which the bills are selected does not matter and seven different types of bills can be selected as many as five times, this problem involves counting 5-combinations with repetition allowed from a set with seven elements. Listing all possibilities would be tedious, since there are a large number of solutions. Instead, we will illustrate the use of a technique for counting combinations with repetition allowed.

Suppose that a cash box has seven compartments, one to hold each type of bill, as illustrated in Figure 1. These bins are separated by six dividers, as shown in the picture. The choice of five bills corresponds to placing five markers in the compartments holding different types of bills. Figure 2 illustrates this correspondence for three different ways to select five bills, where the six dividers are represented by bars and the five bills by stars.

The number of ways to select five bills corresponds to the number of ways to arrange six bars and five stars. Consequently, the number of ways to select the five bills is

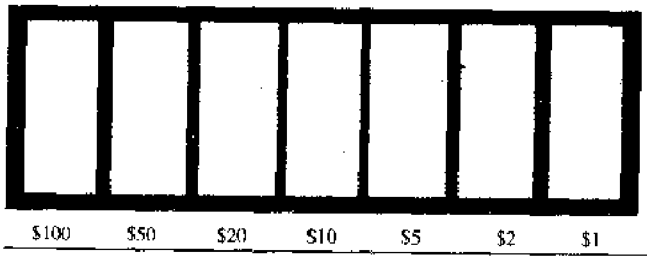


FIGURE 1 Cash Box with Seven Types of Bills.

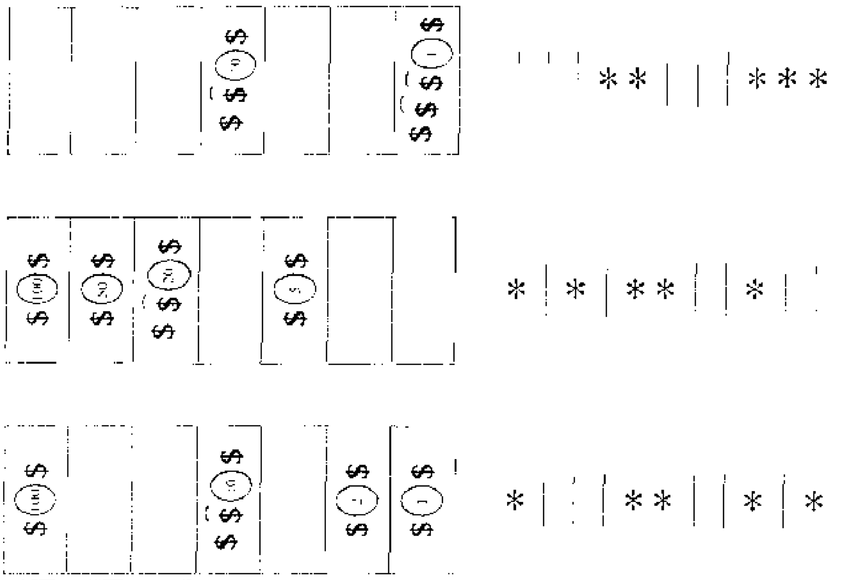


FIGURE 2 **Examples of Ways to Select Five Bills.**

the number of ways to select the positions of the five stars, from 11 possible positions. This corresponds to the number of unordered selections of 5 objects from a set of 11 objects, which can be done in $C(11, 5)$ ways. Consequently, there are

$$C(11,5) = \frac{11!}{5!\,6!} = 462$$

ways to choose five bills from the bag with seven types of bills. ■

The following theorem generalizes this discussion.

THEOREM 2 **There are $C(n + r - 1, r)$ r-combinations from a set with n elements when repetition of elements is allowed.**

Proof: Each r-combination of a set with n elements when repetition is allowed can be represented by a list of $n - 1$ bars and r stars. The $n - 1$ bars are used to mark off n different cells, with the ith cell containing a star for each time the ith element of the set occurs in the combination. For instance, a 6-combination of a set with four elements is represented with three bars and six stars. Here

$$** \mid * \mid \mid ***$$

represents the combination containing exactly two of the first element, one of the second element, none of the third element, and three of the fourth element of the set.

As we have seen, each different list containing $n - 1$ bars and r stars corresponds to an r-combination of the set with n elements, when repetition is allowed. The number of such lists is $C(n - 1 + r, r)$, since each list corresponds to a choice of the r

positions to place the r stars from the $n - 1 + r$ positions that contain r stars and $n - 1$ bars. ☐

The following examples show how Theorem 2 is applied.

EXAMPLE 5 Suppose that a cookie shop has four different kinds of cookies. How many different ways can six cookies be chosen? Assume that only the type of cookie, and not the individual cookies or the order in which they are chosen, matters.

Solution: The number of ways to choose six cookies is the number of 6-combinations of a set with four elements. From Theorem 2 this equals $C(4 + 6 - 1, 6) = C(9, 6)$. Since

$$C(9, 6) = C(9, 3) = \frac{9 \cdot 8 \cdot 7}{1 \cdot 2 \cdot 3} = 84,$$

there are 84 different ways to choose the six cookies. ■

Theorem 2 can also be used to find the number of solutions of certain linear equations where the variables are integers subject to constraints. This is illustrated by the following example.

EXAMPLE 6 *How many solutions does the equation*

$$x_1 + x_2 + x_3 = 11$$

have, where x_1, x_2, and x_3 are nonnegative integers?

Solution: To count the number of solutions, we note that a solution corresponds to a way of selecting 11 items from a set with three elements, so that x_1 items of type one, x_2 items of type two, and x_3 items of type three are chosen. Hence, the number of solutions is equal to the number of 11-combinations with repetition allowed from a set with three elements. From Theorem 2 it follows that there are

$$C(3 + 11 - 1, 11) = C(13, 11) = C(13, 2) = \frac{13 \cdot 12}{1 \cdot 2} = 78$$

solutions.

The number of solutions of this equation can also be found when the variables are subject to constraints. For instance, we can find the number of solutions where the variables are integers with $x_1 \geq 1$, $x_2 \geq 2$, and $x_3 \geq 3$. A solution to the equation subject to these constraints corresponds to a selection of 11 items with x_1 items of type one, x_2 items of type two, and x_3 items of type three, where, in addition, there is at least one item of type one, two items of type two, and three items of type three. So, choose one item of type one, two of type two, and three of type three. Then select five additional items. By Theorem 2 this can be done in

$$C(3 + 5 - 1, 5) = C(7, 5) = C(7, 2) = \frac{7 \cdot 6}{1 \cdot 2} = 21$$

ways. Thus, there are 21 solutions of the equation subject to the given constraints. ■

The following example shows how counting the number of combinations with repetition allowed arises in determining the value of a variable that is incremented each time a certain type of nested loop is traversed.

EXAMPLE 7 What is the value of k after the following pseudocode has been executed?

```
$k := 0$
for $i_1 := 1$ to $n$
  for $i_2 := 1$ to $i_1$
      .
      .
      .
      for $i_m := 1$ to $i_{m-1}$
        $k := k + 1$
```

Solution: Note that the initial value of k is 0 and that 1 is added to k each time the nested loop is traversed with a set of integers $i_1, i_2, \ldots, i_m$ such that

$$1 \le i_m \le i_{m-1} \le \cdots \le i_1 \le n.$$

The number of such sets of integers is the number of ways to choose m integers from $\{1, 2, \ldots, n\}$, with repetition allowed. (To see this, note that once such a set has been selected, if we order the integers in the set in nondecreasing order, this uniquely defines an assignment of $i_m, i_{m-1}, \ldots, i_1$. Conversely, every such assignment corresponds to a unique unordered set.) Hence, from Theorem 2, it follows that $k = C(n + m - 1, m)$ after this code has been executed. ■

The formulae for the numbers of ordered and unordered selections of r elements, chosen with and without repetition allowed from a set with n elements, are shown in Table 1.

TABLE 1 Combinations and Permutations With and Without Repetition.

Type	*Repetition Allowed?*	*Formula*
r-permutations	No	$\frac{n!}{(n-r)!}$
r-combinations	No	$\frac{n!}{r!\,(n-r)!}$
r-permutations	Yes	n^r
r-combinations	Yes	$\frac{(n+r-1)!}{r!\,(n-1)!}$

PERMUTATIONS OF SETS WITH INDISTINGUISHABLE OBJECTS

Some elements may be indistinguishable in counting problems. When this is the case, care must be taken to avoid counting things more than once. Consider the following example.

EXAMPLE 8 How many different strings can be made by reordering the letters of the word

SUCCESS?

Solution: Because some of the letters of *SUCCESS* are the same, the answer is *not* given by the number of permutations of seven letters. This word contains three *S*s, two *C*s, one *U*, and one *E*. To determine the number of different strings that can be made by reordering the letters, first note that the three *S*s can be placed among the seven positions in $C(7, 3)$ different ways, leaving four positions free. Then the two *C*s can be placed in $C(4, 2)$ ways, leaving two free positions. The *U* can be placed in $C(2, 1)$ ways, leaving just one position free. Hence *E* can be placed in $C(1, 1)$ way. Consequently, from the product rule, the number of different strings that can be made is

$$\begin{aligned} C(7,3)C(4,2)C(2,1)C(1,1) &= \frac{7!}{3!\,4!}\cdot\frac{4!}{2!\,2!}\cdot\frac{2!}{1!\,1!}\cdot\frac{1!}{1!\,0!} \\ &= \frac{7!}{3!\,2!\,1!\,1!} \\ &= 420. \end{aligned}$$

■

Using the same sort of reasoning as in the previous example, the following theorem can be proved.

THEOREM 3 **The number of different permutations of n objects, where there are n_1 indistinguishable objects of type 1, n_2 indistinguishable objects of type 2, ..., and n_k indistinguishable objects of type k, is**

$$\frac{n!}{n_1!\,n_2!\cdots n_k!}.$$

Proof: To determine the number of permutations, first note that the n_1 objects of type one can be placed among the n positions in $C(n, n_1)$ ways, leaving $n - n_1$ positions free. Then the objects of type two can be placed in $C(n - n_1, n_2)$ ways, leaving $n - n_1 - n_2$ positions free. Continue placing the objects of type three, ..., type $k - 1$, until at the last stage n_k objects of type k can be placed in $C(n - n_1 - n_2 - \cdots - n_{k-1}, n_k)$ ways. Hence, by the product rule, the total number of different permutations is

$$\begin{aligned} &C(n, n_1)C(n - n_1, n_2)\cdots C(n - n_1 - \cdots - n_{k-1}, n_k) \\ &\qquad = \frac{n!}{n_1!\,(n-n_1)!}\,\frac{(n-n_1)!}{n_2!\,(n-n_1-n_2)!}\cdots\frac{(n-n_1-\cdots-n_{k-1})!}{n_k!\,0!} \\ &\qquad = \frac{n!}{n_1!\,n_2!\cdots n_k!}. \end{aligned}$$

□

DISTRIBUTING OBJECTS INTO BOXES

Some counting problems can be solved by enumerating the ways distinguishable objects can be placed into distinguishable boxes. Consider the following example in which the objects are cards and the "boxes" are hands of players.

EXAMPLE 9 How many ways are there to distribute hands of 5 cards to each of four players from the standard deck of 52 cards?

Solution: We will use the product rule to solve this problem. To begin, note that the first player can be dealt 5 cards in $C(52, 5)$ ways. The second player can be dealt 5 cards in $C(47, 5)$ ways, since only 47 cards are left. The third player can be dealt 5 cards in $C(42, 5)$ ways. Finally, the fourth player can be dealt 5 cards in $C(37, 5)$ ways. Hence, the total number of ways to deal four players 5 cards each is

$$\begin{aligned} C(52,5)C(47,5)C(42,5)C(37,5) &= \frac{52!}{47!\,5!}\cdot\frac{47!}{42!\,5!}\cdot\frac{42!}{37!\,5!}\cdot\frac{37!}{32!\,5!} \\ &= \frac{52!}{5!\,5!\,5!\,5!\,32!}. \end{aligned}$$ ■

Remark: The solution to Example 9 equals the number of permutations of 52 objects, with 5 indistinguishable objects of each of four different types, and 32 objects of a fifth type. This equality can be seen by defining a one-to-one correspondence between permutations of this type and distributions of cards to the players. To define this correspondence, first order the cards from 1 to 52. Then cards dealt to the first player correspond to the cards in the positions assigned to objects of the first type in the permutation. Similarly, cards dealt to the second, third, and fourth players, respectively, correspond to cards in the positions assigned to objects of the second, third, and fourth type, respectively. The cards not dealt to any player correspond to cards in the positions assigned to objects of the fifth type. The reader should verify that this is a one-to-one correspondence.

Example 9 is a typical problem that involves distributing distinguishable objects into distinguishable boxes. The distinguishable objects are the 52 cards, and the five distinguishable boxes are the hands of the four players and the rest of the deck. Counting problems that involve distributing distinguishable objects into boxes can be solved using the following theorem.

THEOREM 4 **The number of ways to distribute n distinguishable objects into k distinguishable boxes so that n_i objects are placed into box i, $i = 1, 2, \ldots, k$, equals**

$$\frac{n!}{n_1!\,n_2!\cdots n_k!}.$$

The proof of Theorem 4 is left for the reader (see Exercises 43 and 44).

Exercises

1. In how many different ways can five elements be selected in order from a set with three elements when repetition is allowed?
2. In how many different ways can five elements be selected in order from a set with five elements when repetition is allowed?
3. How many strings of six letters are there?
4. Every day a student randomly chooses a sandwich for lunch from a pile of wrapped sandwiches. If there are six kinds of sandwiches, how many different ways are there for the student to choose sandwiches for the seven days of a week if the order in which the sandwiches are chosen matters?
5. How many ways are there to assign three jobs to five employees if each employee can be given more than one job?
6. How many ways are there to select five unordered elements from a set with three elements when repetition is allowed?
7. How many ways are there to select three unordered elements from a set with five elements when repetition is allowed?
8. How many different ways are there to choose a dozen donuts from the 21 varieties at a donut shop?
9. A bagel shop has onion bagels, poppy seed bagels, egg bagels, salty bagels, pumpernickel bagels, sesame seed bagels, raisin bagels, and plain bagels. How many ways are there to choose
   a) six bagels?
   b) a dozen bagels?
   c) two dozen bagels?
   d) a dozen bagels with at least one of each kind?
   e) a dozen bagels with at least three egg bagels and no more than two salty bagels?
10. A croissant shop has plain croissants, cherry croissants, chocolate croissants, almond croissants, apple croissants, and broccoli croissants. How many ways are there to choose
    a) a dozen croissants?
    b) three dozen croissants?
    c) two dozen croissants with at least two of each kind?
    d) two dozen croissants with no more than two broccoli croissants?
    e) two dozen croissants with at least five chocolate croissants and at least three almond croissants?
    f) two dozen croissants with at least one plain croissant, at least two cherry croissants, at least three chocolate croissants, at least one almond croissant, at least two apple croissants, and no more than three broccoli croissants?
11. How many ways are there to choose eight coins from a piggy bank containing 100 identical pennies and 80 identical nickels?
12. How many different combinations of pennies, nickels, dimes, quarters, and half dollars can a piggy bank contain if it has 20 coins in it?
13. A book publisher has 3000 copies of a discrete mathematics book. How many ways are there to store these books in their three warehouses if the copies of the book are indistinguishable?
14. How many solutions are there to the equation

    $$x_1 + x_2 + x_3 + x_4 = 17$$

    where x_1, x_2, x_3, and x_4 are nonnegative integers?
15. How many solutions are there to the equation

    $$x_1 + x_2 + x_3 + x_4 + x_5 = 21$$

    where x_i, $i = 1, 2, 3, 4, 5$, is a nonnegative integer such that
    a) $x_1 \geq 1$?
    b) $x_i \geq 2$ for $i = 1, 2, 3, 4, 5$?
    c) $0 \leq x_1 \leq 10$?
    d) $0 \leq x_1 \leq 3$, $1 \leq x_2 < 4$, and $x_3 \geq 15$?
16. How many solutions are there to the equation

    $$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 = 29$$

    where x_i, $i = 1, 2, 3, 4, 5, 6$, is a nonnegative integer such that
    a) $x_i > 1$ for $i = 1, 2, 3, 4, 5, 6$?
    b) $x_1 \geq 1$, $x_2 \geq 2$, $x_3 \geq 3$, $x_4 \geq 4$, $x_5 > 5$, and $x_6 \geq 6$?
    c) $x_1 \leq 5$?
    d) $x_1 < 8$ and $x_2 > 8$?
17. How many strings of 10 ternary digits (0, 1, or 2) are there that contain exactly two 0s, three 1s, and five 2s?
18. How many strings of 20 decimal digits are there that contain two 0s, four 1s, three 2s, one 3, two 4s, three 5s, two 7s, and three 9s?
19. Suppose that a large family has 14 children, including two sets of identical triplets, three sets of identical twins, and two individual children. How many ways are there to seat these children in a row of chairs if the identical triplets or twins cannot be distinguished from one another?
20. How many solutions are there to the inequality

    $$x_1 + x_2 + x_3 \leq 11$$

    where x_1, x_2, and x_3 are nonnegative integers? (*Hint:* Introduce an auxiliary variable x_4 so that $x_1 + x_2 + x_3 + x_4 = 11$.)

21. How many positive integers less than 1,000,000 have the sum of their digits equal to 19?

22. How many positive integers less than 1,000,000 have exactly one digit equal to 9 and have a sum of digits equal to 13?

23. There are 10 questions on a discrete mathematics final exam. How many ways are there to assign scores to the problems if the sum of the scores is 100 and each question is worth at least 5 points?

24. Show that there are $C(n + r - q_1 - q_2 - \cdots - q_r - 1, n - q_1 - q_2 - \cdots - q_r)$ different unordered selections of n objects of r different types that include at least q_1 objects of type one, q_2 objects of type two, ..., and q_r objects of type r.

25. How many different bit strings can be transmitted if the string must begin with a 1 bit, must include three additional 1 bits (so that a total of four 1 bits is sent), must include a total of twelve 0 bits, and must have at least two 0 bits following each 1 bit?

26. How many different strings can be made from the letters in *MISSISSIPPI*, using all the letters?

27. How many different strings can be made from the letters in *ABRACADABRA*, using all the letters?

28. How many different strings can be made from the letters in *AARDVARK*, using all the letters, if all three *A*s must be consecutive?

29. How many different strings can be made from the letters in *ORONO*, using some or all of the letters?

30. How many strings with five or more characters can be formed from the letters in *SEERESS*?

31. How many strings with seven or more characters can be formed from the letters in *EVERGREEN*?

32. How many different bit strings can be formed using six 1s and eight 0s?

33. A student has three mangos, two papayas, and two kiwi fruits. If the student eats one piece of fruit each day, and only the type of fruit matters, in how many different ways can these fruits be consumed?

34. A professor packs her collection of 40 issues of a mathematics journal in four boxes with 10 issues per box. How many ways can she distribute the journals if
  a) each box is numbered, so that they are distinguishable?
  b) the boxes are identical, so that they cannot be distinguished?

35. How many ways are there to travel in xyz space from the origin $(0, 0, 0)$ to the point $(4, 3, 5)$ by taking steps one unit in the positive x direction, one unit in the positive y direction, or one unit in the positive z direction? (Moving in the negative x, y, or z direction is prohibited, so that no backtracking is allowed.)

36. How many ways are there to travel in $xyzw$ space from the origin $(0, 0, 0, 0)$ to the point $(4, 3, 5, 4)$ by taking steps one unit in the positive x, positive y, positive z, or positive w direction?

37. How many ways are there to deal hands of 7 cards to each of five players from a standard deck of 52 cards?

38. In bridge, the 52 cards of a standard deck are dealt to four players. How many different ways are there to deal bridge hands to four players?

39. What is the probability that each player has a hand containing an ace when the 52 cards of a standard deck are dealt to four players?

40. In how many ways can a dozen books be placed on four distinguishable shelves
  a) if the books are indistinguishable copies of the same title?
  b) if no two books are the same, and the positions of the books on the shelves matter? (*Hint:* Break this into 12 tasks, placing each book separately. Start with the sequence 1, 2, 3, 4 to represent the shelves. Represent the books by b_i, $i = 1, 2, \ldots, 12$. Place b_1 to the right of one of the terms in 1, 2, 3, 4. Then successively place $b_2, b_3, \ldots,$ and b_{12}.)

41. How many ways can n books be placed on k distinguishable shelves
  a) if the books are indistinguishable copies of the same title?
  b) if no two books are the same, and the positions of the books on the shelves matter?

42. A shelf holds 12 books in a row. How many ways are there to choose five books so that no two adjacent books are chosen? (*Hint:* Represent the books that are chosen by bars and the books not chosen by stars. Count the number of sequences of five bars and seven stars so that no two bars are adjacent.)

***43.** Use the product rule to prove Theorem 4, by first placing objects in the first box, then placing objects in the second box, and so on.

***44.** Prove Theorem 4 by first setting up a one-to-one correspondence between permutations of n objects with n_i indistinguishable objects of type i, $i = 1, 2, 3, \ldots, k$, and the distributions of n objects in k boxes such that n_i objects are placed in box i, $i = 1, 2, 3, \ldots, k$ and then applying Theorem 3.

***45.** In this exercise we will prove Theorem 2 by setting up a one-to-one correspondence between the set of r-combinations with repetition allowed of $S = \{1, 2, 3, \ldots, n\}$ and the set of r-combinations of the set $T = \{1, 2, 3, \ldots, n + r - 1\}$.
  a) Arrange the elements in an r-combination, with repetition allowed, of S into an increasing sequence $x_1 \le x_2 \le \cdots \le x_r$. Show that the sequence formed by adding $k - 1$ to the kth term is strictly increasing. Conclude that this sequence is made up of r distinct elements from T.
  b) Show that the procedure described in (a) defines a one-to-one correspondence between the set of r-combinations, with repetition allowed, of S and

the r-combinations of T. (*Hint:* Show the correspondence can be reversed by associating to the r-combination $\{x_1, x_2, \ldots, x_r\}$ of T, with $1 \le x_1 < x_2 < \cdots < x_r \le n + r - 1$, the r-combination with repetition allowed from S, formed by subtracting $k - 1$ from the kth element.)

c) Conclude that there are $C(n + r - 1, r)$ r-combinations with repetition allowed from a set with n elements.

46. How many ways are there to distribute five distinguishable objects into three indistinguishable boxes?

47. How many ways are there to distribute five indistinguishable objects into three indistinguishable boxes?

48. How many different terms are there in the expansion of $(x_1 + x_2 + \cdots + x_m)^n$ after all terms with identical sets of exponents are added?

***49.** Prove the **multinomial theorem:** If n is a positive integer, then

$$(x_1 + x_2 + \cdots + x_m)^n$$
$$= \sum_{n_1 + n_2 + \cdots + n_m = n} C(n; n_1, n_2, \ldots, n_m) x_1^{n_1} x_2^{n_2} \cdots x_n^{n_m},$$

where

$$C(n; n_1, n_2, \ldots, n_m) = \frac{n!}{n_1!\, n_2! \cdots n_m!}$$

is a **multinomial coefficient.**

50. Find the expansion of $(x + y + z)^4$.

51. Find the coefficient of $x^3 y^2 z^5$ in $(x + y + z)^{10}$.

52. How many terms are there in the expansion of $(x + y + z)^{100}$?

4.7

Generating Permutations and Combinations

INTRODUCTION

Methods for counting various types of permutations and combinations were described in the previous sections of this chapter, but sometimes permutations or combinations need to be generated, not just counted. Consider the following three problems. First, suppose that a salesman must visit six different cities. In which order should these cities be visited to minimize total travel time? One way to determine the best order is to determine the travel time for each of the $6! = 720$ different orders in which the cities can be visited and choose the one with the smallest travel time. Second, suppose some numbers from a set of six numbers have 100 as their sum. One way to find these numbers is to generate all the $2^6 = 64$ subsets and check the sum of their terms. Third, suppose a laboratory has 95 employees. A group of 12 of these employees with a particular set of 25 skills is needed for a project. (Each employee can have one or more of these skills.) One way to find such a set of employees is to generate all sets of 12 of these employees and check whether they have the desired skills. These examples show that it is often necessary to generate permutations and combinations to solve problems.

GENERATING PERMUTATIONS

web

Any set with n elements can be placed in one-to-one correspondence with the set $\{1, 2, 3, \ldots, n\}$. We can list the permutations of any set of n elements by generating the permutations of the n smallest positive integers and then replacing these integers with the corresponding elements. Many different algorithms have been developed to generate the $n!$ permutations of this set. We will describe one of these that is based on the **lexicographic ordering** of the set of permutations of $\{1, 2, 3, \ldots, n\}$. In this ordering, the permutation $a_1 a_2 \cdots a_n$ precedes the permutation of $b_1 b_2 \cdots b_n$, if for some k, with $1 \le k \le n$, $a_1 = b_1, a_2 = b_2, \ldots, a_{k-1} = b_{k-1}$, and $a_k < b_k$. In other words, a permutation of the set of the n smallest positive integers precedes (in lexicographic order) a second permutation if the number in this permutation in the first position where the two permutations disagree is smaller than the number in that position in the second permutation.

EXAMPLE 1 The permutation 23415 of the set $\{1, 2, 3, 4, 5\}$ precedes the permutation 23514, since these permutations agree in the first two positions, but the number in the third position in the first permutation, 4, is smaller than the number in the third position in the second permutation, 5. Similarly, the permutation 41532 precedes 52143. ■

An algorithm for generating the permutations of $\{1, 2, \ldots, n\}$ can be based on a procedure that constructs the next permutation in lexicographic order following a given permutation $a_1a_2 \cdots a_n$. We will show how this can be done. First, suppose that $a_{n-1} < a_n$. Interchange a_{n-1} and a_n to obtain a larger permutation. No other permutation is both larger than the original permutation and smaller than the permutation obtained by interchanging a_{n-1} and a_n. For instance, the next largest permutation after 234156 is 234165. On the other hand, if $a_{n-1} > a_n$, then a larger permutation cannot be obtained by interchanging these last two terms in the permutation. Look at the last three integers in the permutation. If $a_{n-2} < a_{n-1}$, then the last three integers in the permutation can be rearranged to obtain the next largest permutation. Put the smaller of the two integers a_{n-1} and a_n that is greater than a_{n-2} in position $n-2$. Then, place the remaining integer and a_{n-2} into the last two positions in increasing order. For instance, the next largest permutation after 234165 is 234516.

On the other hand, if $a_{n-2} > a_{n-1}$ (and $a_{n-1} > a_n$), then a larger permutation cannot be obtained by permuting the last three terms in the permutation. Based on these observations, a general method can be described for producing the next largest permutation in increasing order following a given permutation $a_1a_2 \cdots a_n$. First, find the integers a_j and a_{j+1} with $a_j < a_{j+1}$ and

$$a_{j+1} > a_{j+2} > \cdots > a_n,$$

that is, the last pair of adjacent integers in the permutation where the first integer in the pair is smaller than the second. Then, the next largest permutation in lexicographic order is obtained by putting in the jth position the least integer among $a_{j+1}, a_{j+2}, \ldots,$ and a_n that is greater than a_j and listing in increasing order the rest of the integers $a_j, a_{j+1}, \ldots, a_n$ in positions $j+1$ to n. It is easy to see that there is no other permutation larger than the permutation $a_1a_2 \cdots a_n$ but smaller than the new permutation produced. (The verification of this fact is left as an exercise for the reader.)

EXAMPLE 2 What is the next largest permutation in lexicographic order after 362541?

Solution: The last pair of integers a_j and a_{j+1} where $a_j < a_{j+1}$ is $a_3 = 2$ and $a_4 = 5$. The least integer to the right of 2 that is greater than 2 in the permutation is $a_5 = 4$. Hence, 4 is placed in the third position. Then the integers 2, 5, and 1 are placed in order in the last three positions, giving 125 as the last three positions of the permutation. Hence, the next permutation is 364125. ■

To produce the $n!$ permutations of the integers $1, 2, 3, \ldots, n$, begin with the smallest permutation in lexicographic order, namely, $123 \cdots n$, and successively apply the procedure described for producing the next largest permutation of $n! - 1$ times. This yields all the permutations of the n smallest integers in lexicographic order.

EXAMPLE 3 Generate the permutations of the integers 1, 2, 3 in lexicographic order.

Solution: Begin with 123. The next permutation is obtained by interchanging 3 and 2 to obtain 132. Next, since $3 > 2$ and $1 < 3$, permute the three integers in 132. Put the smaller of 3 and 2 in the first position, and then put 1 and 3 in increasing order in positions 2 and 3 to obtain 213. This is followed by 231, obtained by interchanging 1 and 3, since $1 < 3$. The next largest permutation has 3 in the first position, followed by 1 and 2 in increasing order, namely, 312. Finally, interchange 1 and 2 to obtain the last permutation, 321. ■

Algorithm 1 displays the procedure for finding the next largest permutation in lexicographic order after a permutation that is not $n\ n-1\ n-2\ \ldots\ 2\ 1$, which is the largest permutation.

ALGORITHM 1 Generating the Next Largest Permutation in Lexicographic Order.

```
procedure next permutation($a_1a_2 \cdots a_n$: permutation of
   $\{1, 2, \ldots, n\}$ not equal to $n\ \ n-1\ \ \cdots\ \ 2\ \ 1$)
$j := n - 1$
while $a_j > a_{j+1}$
   $j := j - 1$
{$j$ is the largest subscript with $a_j < a_{j+1}$}
$k := n$
while $a_j > a_k$
   $k := k - 1$
{$a_k$ is the smallest integer greater than $a_j$ to the right of $a_j$}
interchange $a_j$ and $a_k$
$r := n$
$s := j + 1$
while $r > s$
begin
   interchange $a_r$ and $a_s$
   $r := r - 1$
   $s := s + 1$
end
{this puts the tail end of the permutation after the $j$th position in
   increasing order}
```

GENERATING COMBINATIONS

web

How can we generate all the combinations of the elements of a finite set? Since a combination is just a subset, we can use the correspondence between subsets of $\{a_1, a_2, \ldots, a_n\}$ and bit strings of length n.

Recall that the bit string corresponding to a subset has a 1 in position k if a_k is in the subset, and has a 0 in this position if a_k is not in the subset. If all the bit strings of length n can be listed, then by the correspondence between subsets and bit strings, a list of all the subsets is obtained.

Recall that a bit string of length n is also the binary expansion of an integer between 0 and $2^n - 1$. The 2^n bit strings can be listed in order of their increasing size as integers in their binary expansions. To produce all binary expansions of length n, start with the bit string 000...00, with n zeros. Then, successively find the next largest expansion until the bit string 111...11 is obtained. At each stage the next largest binary expansion is found by locating the first position from the right that is not a 1, then changing all the 1s to the right of this position to 0s and making this first 0 (from the right) a 1.

EXAMPLE 4 Find the next largest bit string after 10 0010 0111.

Solution: The first bit from the right that is not a 1 is the fourth bit from the right. Change this bit to a 1 and change all the following bits to 0s. This produces the next largest bit string, 10 0010 1000. ■

The procedure for producing the next largest bit string after $b_{n-1}b_{n-2}\ldots b_1b_0$ is given as Algorithm 2.

ALGORITHM 2 Generating the Next Largest Bit String.

```
procedure next bit string($b_{n-1}b_{n-2}\ldots b_1b_0$: bit string not equal to 11...11)
i := 0
while $b_i = 1$
begin
  $b_i := 0$
  $i := i + 1$
end
$b_i := 1$
```

Next, an algorithm for generating the r-combinations of the set $\{1, 2, 3, \ldots, n\}$ will be given. An r-combination can be represented by a sequence containing the elements in the subset in increasing order. The r-combinations can be listed using lexicographic order on these sequences. The next combinations after $a_1a_2 \cdots a_r$ can be obtained in the following way: First, locate the last element a_i in the sequence such that $a_i \neq n - r + i$. Then, replace a_i with $a_i + 1$ and a_j with $a_i + j - i + 1$, for $j = i + 1, i + 2, \ldots, r$. It is left for the reader to show that this produces the next largest combination in lexicographic order. This procedure is illustrated with the following example.

EXAMPLE 5 Find the next largest 4-combination of the set $\{1, 2, 3, 4, 5, 6\}$ after $\{1, 2, 5, 6\}$.

Solution: The last term among the terms a_i with $a_1 = 1$, $a_2 = 2$, $a_3 = 5$, and $a_4 = 6$ such that $a_i \neq 6 - 4 + i$ is $a_2 = 2$. To obtain the next largest 4-combination, increment

a_2 by 1 to obtain $a_2 = 3$. Then set $a_3 = 3 + 1 = 4$ and $a_4 = 3 + 2 = 5$. Hence the next largest 4-combination is $\{1, 3, 4, 5\}$. ■

Algorithm 3 gives this procedure in pseudocode.

ALGORITHM 3 Generating the Next *r*-Combination in Lexicographic Order.

```
procedure next r-combination($\{a_1, a_2, \ldots, a_r\}$: proper subset of
    $\{1, 2, \ldots, n\}$ not equal to $\{n - r + 1, \ldots, n\}$ with
    $a_1 < a_2 < \cdots < a_r$)
$i := r$
while $a_i = n - r + i$
    $i := i - 1$
$a_i := a_i + 1$
for $j := i + 1$ to $r$
    $a_j := a_i + j - i$
```

Exercises

1. Find the next largest permutation in lexicographic order after each of the following permutations.
a) 1432 b) 54123
c) 12453 d) 45231
e) 6714235 f) 31528764

2. Place the following permutations of $\{1, 2, 3, 4, 5, 6\}$ in lexicographic order: 234561, 231456, 165432, 156423, 543216, 541236, 231465, 314562, 432561, 654321, 654312, 435612.

3. Use Algorithm 1 to generate the 24 permutations of the first four positive integers in lexicographic order.

4. Use Algorithm 2 to list all the subsets of the set $\{1, 2, 3, 4\}$.

5. Use Algorithm 3 to list all the 3-combinations of $\{1, 2, 3, 4, 5\}$.

6. Show that Algorithm 1 produces the next largest permutation in lexicographic order.

7. Show that Algorithm 3 produces the next largest r-combination in lexicographic order after a given r-combination.

8. Develop an algorithm for generating the r-permutations of a set of n elements.

9. List all 3-permutations of $\{1, 2, 3, 4, 5\}$.

The remaining exercises in this section develop another algorithm for generating the permutations of $\{1, 2, 3, \ldots, n\}$. This algorithm is based on Cantor expansions of integers. Every nonnegative integer less than $n!$ has a unique Cantor expansion

$$a_1 1! + a_2 2! + \cdots + a_{n-1}(n-1)!$$

where a_i is a nonnegative integer not exceeding i, for $i = 1, 2, \ldots, n - 1$. The integers $a_1, a_2, \ldots, a_{n-1}$ are called the **Cantor digits** of this integer.

Given a permutation of $\{1, 2, \ldots, n\}$, let a_{k-1}, $k = 2, 3, \ldots, n$, be the number of integers less than k that follow k in the permutation. For instance, in the permutation 43215, a_1 is the number of integers less than 2 that follow 2, so that $a_1 = 1$. Similarly, for this example $a_2 = 2$, $a_3 = 3$, and $a_4 = 0$. Consider the function from the set of permutations $\{1, 2, 3, \ldots, n\}$ to the set of nonnegative integers less than $n!$ that sends a permutation to the integer that has $a_1, a_2, \ldots, a_{n-1}$, defined in this way, as its Cantor digits.

10. Find the integers that correspond to the following permutations.
a) 246531 b) 12345 c) 654321

***11.** Show that the correspondence described here is a bijection between the set of permutations of $\{1, 2, 3, \ldots, n\}$ and the nonnegative integers less than $n!$.

12. Find the permutations of $\{1, 2, 3, 4, 5\}$ that correspond to the following integers with respect to the correspondence between Cantor expansions and permutations as described before Exercise 10.
a) 3 b) 89 c) 111

13. Develop an algorithm for producing all permutations of a set of n elements based on the correspondence described in the preamble to Exercise 10.

***14.** The following method can be used to generate a random permutation of a sequence of n terms. First, interchange the nth term and the $r(n)$th term where $r(n)$ is a randomly selected integer with $1 \leq r(n) \leq n$. Next, interchange the $(n-1)$th term of the resulting sequence with its $r(n-1)$th term where $r(n-1)$ is a randomly selected integer with $1 \leq r(n-1) \leq n-1$. Continue this process until $j = n$, where at the jth step you interchange the $(n-j+1)$th term of the resulting sequence with its $r(n-j+1)$th term, where $r(n-j+1)$ is a randomly selected integer with $1 \leq r(n-j+1) \leq n-j+1$. Show that when this method is followed each of the $n!$ different permutations of the terms of the sequence is equally likely to be generated. [*Hint:* Use mathematical induction, assuming that the probability that each of the permutations of $n-1$ terms produced by this procedure for a sequence of $n-1$ terms is $1/(n-1)!$.]

Key Terms and Concepts

TERMS

combinatorics: the study of arrangements of objects

enumeration: the counting of arrangements of objects

tree diagram: a diagram made up of a root, branches leaving the root, and other branches leaving some of the endpoints of branches

permutation: an ordered arrangement of the elements of a set

r-permutation: an ordered arrangement of r elements of a set

$P(n, r)$: the number of r-permutations of a set with n elements

r-combination: an unordered selection of r elements of a set

$C(n, r)$: the number of r-combinations of a set with n elements

$\binom{n}{r}$ (binomial coefficient): also the number of r-combinations of a set with n elements

Pascal's triangle: a representation of the binomial coefficients where the ith row of the triangle contains $C(i, j)$ for $j = 0, 1, 2, \ldots, i$

probability of an event: the number of successful outcomes of this event divided by the number of possible outcomes

$p(E \mid F)$ (conditional probability of E given F): $p(E \cap F)/p(F)$

independent events: events E and F such that $p(E \cap F) = p(E)p(F)$

random variable: a function that assigns a real number to each outcome of an experiment

expected value of a random variable: the weighted average of a random variable, with values of the random variable weighted by the probability of outcomes, that is, $E(X) = \sum_{s \in S} p(s)X(s)$

variance of a random variable: the weighted average of the square of the difference between the value of the random variable and its expected value, with weights given by the probability of outcomes, that is, $V(X) = \sum_{s \in S}(X(s) - E(X))^2 p(s)$

Bernoulli trial: an experiment with two possible outcomes

RESULTS

The sum rule: a basic counting technique which states that the number of ways to do a task in one of two ways is the sum of the number of ways to do these tasks if they cannot be done simultaneously

The product rule: a basic counting technique which states that the number of ways to do a procedure that consists of two subtasks is the product of the number of ways to do the first task and the number of ways to do the second task after the first task has been done

The pigeonhole principle: When more than k objects are placed in k boxes, there must be a box containing more than one object.

The generalized pigeonhole principle: When N objects are placed in k boxes, there must be a box containing at least $\lceil N/k \rceil$ objects.

$$P(n, r) = \frac{n!}{(n-r)!}$$

$$C(n, r) = \frac{n!}{r!\,(n-r)!}$$

Pascal's identity: $C(n+1, k) = C(n, k-1) + C(n, k)$

The binomial theorem: $(x+y)^n = \sum_{k=0}^{n} C(n, k)x^{n-k}y^k$

The probability of k successes when n independent Bernoulli trials are carried out equals $C(n, k)p^k q^{n-k}$, where p is the probability of success and $q = 1 - p$ is the probability of failure.

There are n^r r-permutations of a set with n elements when repetition is allowed.

There are $C(n + r - 1, r)$ r-combinations of a set with n elements when repetition is allowed.

There are $n!/(n_1!\,n_2!\cdots n_k!)$ permutations of n objects where there are n_i indistinguishable objects of type i for $i = 1, 2, 3, \ldots, k$.

The algorithm for generating the permutations of the set $\{1, 2, \ldots, n\}$.

Review Questions

1. Explain how the sum and product rules can be used to find the number of bit strings with a length not exceeding 10.

2. Explain how to find the number of bit strings of length not exceeding 10 that have at least one 0 bit.

3. **a)** How can the product rule be used to find the number of functions from a set with m elements to a set with n elements?
  b) How many functions are there from a set with 5 elements to a set with 10 elements?
  c) How can the product rule be used to find the number of one-to-one functions from a set with m elements to a set with n elements?
  d) How many one-to-one functions are there from a set with 5 elements to a set with 10 elements?
  e) How many onto functions are there from a set with 5 elements to a set with 10 elements?

4. How can you find the number of possible outcomes of a playoff between two teams where the first team that wins four games wins the playoff?

5. How can you find the number of bit strings of length 10 that either begin with 101 or end with 010?

6. **a)** State the pigeonhole principle.
  b) Explain how the pigeonhole principle can be used to show that among any 11 integers, at least 2 must have the same last digit.

7. **a)** State the generalized pigeonhole principle.
  b) Explain how the generalized pigeonhole principle can be used to show that among any 91 integers, there are at least 10 that end with the same digit.

8. **a)** What is the difference between an r-combination and an r-permutation of a set with n elements?
  b) Derive an equation that relates the number of r-combinations and the number of r-permutations of a set with n elements.
  c) How many ways are there to select 6 students from a class of 25 to serve on a committee?
  d) How many ways are there to select 6 students from a class of 25 to hold six different executive positions on a committee?

9. **a)** What is Pascal's triangle?
  b) How can a row of Pascal's triangle be produced from the one above it?

10. What is meant by a combinatorial proof of an identity? How is such a proof different from an algebraic one?

11. Explain how to prove Pascal's identity using a combinatorial argument.

12. **a)** State the binomial theorem.
  b) Explain how to prove the binomial theorem using a combinatorial argument.
  c) Find the coefficient of $x^{100}y^{101}$ in the expansion of $(2x + 5y)^{201}$.

13. **a)** Define the probability of an event when all outcomes are equally likely.
  b) What is the probability that you select the six winning numbers in a lottery if the six different winning numbers are selected from the first 50 positive integers?

14. **a)** What conditions should be met by the probabilities assigned to the outcomes from a finite sample space?
  b) What probabilities should be assigned to the outcome of heads and the outcome of tails if heads comes up three times as often as tails?

15. **a)** Define the conditional probability of an event E given an event F.
  b) Suppose E is the event that when a die is rolled it comes up an even number, and F is the event that when a die is rolled it comes up 1, 2, or 3. What is the probability of F given E?

16. **a)** When are two events E and F independent?
  b) Suppose E is the event that an even number appears when a fair die is rolled, and F is the event that a 5 or 6 comes up. Are E and F independent?

17. **a)** What is a random variable?
  b) What are the values assigned by the random variable X that assigns to a roll of two dice the larger number that appears on the two dice?

18. **a)** Define the expected value of a random variable X.
  b) What is the expected value of the random variable X that assigns to a roll of two dice the larger number that appears on the two dice?

19. **a)** Explain how the average-case computational complexity of an algorithm, with finitely many possible input values, can be interpreted as an expected value.
  b) What is the average-case computational complexity of the linear search algorithm, if the probability that the element for which we search is in the list is 1/3, and it is equally likely that this element is any of the n elements in the list?

20. **a)** What is meant by a Bernoulli trial?
  b) What is the probability of k successes in n independent Bernoulli trials?
  c) What is the expected value of the number of successes in n independent Bernoulli trials?

21. **a)** What is the variance of a random variable?
  b) What is the variance of a Bernoulli trial with probability p of success?

22. **a)** What is the variance of the sum of n independent random variables?

b) What is the variance of the number of successes when n independent Bernoulli trials, each with probability p of success, are carried out?

23. **a)** Explain how to find a formula for the number of ways to select r objects from n objects when repetition is allowed and order does not matter.

b) How many ways are there to select a dozen objects from among objects of five different types if objects of the same type are indistinguishable?

c) How many ways are there to select a dozen objects from these five different types if there must be at least three objects of the first type?

d) How many ways are there to select a dozen objects from these five different types if there cannot be more than four objects of the first type?

e) How many ways are there to select a dozen objects from these five different types if there must be at least two objects of the first type but no more than three objects of the second type?

24. **a)** Let n and r be positive integers. Explain why the number of solutions of the equation $x_1 + x_2 + \cdots + x_n = r$, where x_i is a nonnegative integer for $i = 1, 2, 3, \ldots, n$, equals the number of r-combinations of a set with n elements.

b) How many solutions in nonnegative integers are there to the equation $x_1 + x_2 + x_3 + x_4 = 17$?

c) How many solutions in positive integers are there to the equation in part (b)?

25. **a)** Derive a formula for the number of permutations of n objects of k different types where there are n_1 indistinguishable objects of type one, n_2 indistinguishable objects of type two, …, and n_k indistinguishable objects of type k.

b) How many ways are there to order the letters of the word *INDISCREETNESS*?

26. Describe an algorithm for generating all the permutations of the set of the n smallest positive integers.

27. **a)** How many ways are there to deal hands of 5 cards to six players from a standard 52-card deck?

b) How many ways are there to distribute n distinguishable objects into k distinguishable boxes so that n_i objects are placed in box i?

28. Describe an algorithm for generating all the combinations of the set of the n smallest positive integers.

Supplementary Exercises

1. How many ways are there to choose 6 items from 10 distinct items when

a) the items in the choices are ordered and repetition is not allowed?

b) the items in the choices are ordered and repetition is allowed?

c) the items in the choices are unordered and repetition is not allowed?

d) the items in the choices are unordered and repetition is allowed?

2. How many ways are there to choose 10 items from 6 distinct items when

a) the items in the choices are ordered and repetition is not allowed?

b) the items in the choices are ordered and repetition is allowed?

c) the items in the choices are unordered and repetition is not allowed?

d) the items in the choices are unordered and repetition is allowed?

3. A test contains 100 true/false questions. How many different ways can a student answer the questions on the test, if answers may be left blank?

4. How many bit strings of length 10 either start with 000 or end with 1111?

5. How many bit strings of length 10 over the alphabet $\{a, b, c\}$ have either exactly three *a*s or exactly four *b*s?

6. The internal telephone numbers in the phone system on a campus consist of five digits, with the first digit not equal to zero. How many different numbers can be assigned in this system?

7. An ice cream parlor has 28 different flavors, 8 different kinds of sauce, and 12 toppings.

a) In how many different ways can a dish of three scoops of ice cream be made where each flavor can be used more than once and the order of the scoops does not matter?

b) How many different kinds of small sundaes are there if a small sundae contains one scoop of ice cream, a sauce, and a topping?

c) How many different kinds of large sundaes are there if a large sundae contains three scoops of ice cream, where each flavor can be used more than once and the order of the scoops does not matter; two kinds of sauce, where each sauce can be used only once and the order of the sauces does not matter; and three toppings, where each topping can be used only once and the order of the toppings does not matter?

8. How many positive integers less than 1000

a) have exactly three decimal digits?

b) have an odd number of decimal digits?

c) have at least one decimal digit equal to 9?

d) have no odd decimal digits?
e) have two consecutive decimal digits equal to 5?
f) are palindromes (that is, read the same forward and backward)?

9. When the numbers from 1 to 1000 are written out in decimal notation, how many of the following digits are used?
a) 0 b) 1 c) 2 d) 9

10. There are 12 signs of the zodiac. How many people are needed to guarantee that at least six of these people have the same sign?

11. A fortune cookie company makes 213 different fortunes. A student eats at a restaurant that uses fortunes from this company. What is the largest possible number of times that the student can eat at the restaurant without getting the same fortune four times?

12. How many people are needed to guarantee that at least two were born on the same day of the week and in the same month (perhaps in different years)?

13. Show that there are at least two different five-element subsets of a set of 10 positive integers not exceeding 50 that have the same sum.

14. A package of baseball cards contains 20 cards. How many packages must be purchased to ensure that two cards in these packages are identical if there are a total of 550 different cards?

15. a) How many cards must be chosen from a deck to guarantee that at least two aces are chosen?
b) How many cards must be chosen from a deck to guarantee that at least two aces and two kinds are chosen?
c) How many cards must be chosen from a deck to guarantee that there are at least two cards of the same kind?
d) How many cards must be chosen from a deck to guarantee that there are at least two cards of two different kinds?

***16.** Show that in any set of $n + 1$ positive integers not exceeding $2n$ there must be two that are relatively prime.

***17.** Show that in a sequence of m integers there exists one or more consecutive terms with a sum divisible by m.

18. Show that if five points are picked in the interior of a square with a side length of 2, then at least two of these points are no farther than $\sqrt{2}$ apart.

19. Show that the decimal expansion of a rational number must repeat itself from some point onward.

20. How many diagonals does a regular polygon with n sides have, where n is a positive integer with $n \geq 3$?

21. How many ways are there to choose a dozen donuts from 20 varieties
a) if there are no two donuts of the same variety?
b) if all donuts are of the same variety?
c) if there are no restrictions?
d) if there are at least two varieties?
e) if there must be at least six blueberry-filled donuts?
f) if there can be no more than six blueberry-filled donuts?

22. What is the probability that six consecutive numbers will be chosen as the winning numbers in a lottery where each number chosen is between 1 and 40 (inclusive)?

23. What is the probability that a hand of 13 cards contains no pairs?

24. Find n if
a) $P(n, 2) = 110$. b) $P(n, n) = 5040$.
c) $P(n, 4) = 12P(n, 2)$.

25. Find n if
a) $C(n, 2) = 45$. b) $C(n, 3) = P(n, 2)$.
c) $C(n, 5) = C(n, 2)$.

26. Show that if n and r are nonnegative integers and $n \geq r$, then

$$P(n + 1, r) = P(n, r)(n + 1)/(n + 1 - r).$$

27. Give a combinatorial proof that $C(n, r) = C(n, n - r)$.

28. Give a combinatorial proof of Theorem 7 of Section 4.3 by setting up a correspondence between the subsets of a set with an even number of elements and the subsets of this set with an odd number of elements. (*Hint:* Take an element a in the set. Set up the correspondence by putting a in the subset if it is not already in it and taking it out if it is in the subset.)

29. Let n and r be nonnegative integers with $r < n$. Show that

$$C(n, r - 1) = C(n + 2, r + 1) - 2C(n + 1, r + 1) + C(n, r + 1).$$

30. Prove using mathematical induction that $\sum_{j=2}^{n} C(j, 2) = C(n + 1, 3)$ whenever n is an integer greater than 1.

31. Use the binomial theorem to prove that $3^n = \sum_{k=0}^{n} C(n, k)2^k$. (*Hint:* Let $x = 1$ and $y = 2$ in the statement of the theorem.)

32. *In this exercise we will derive a formula for the sum of* the squares of the n smallest positive integers. We will count the number of triples (i, j, k) such that i, j, and k are integers such that $0 \leq i < k$, $0 \leq j < k$, and $1 \leq k \leq n$ in two ways.
a) Show that there are k^2 such triples with a fixed k. Conclude that there are $\sum_{k=1}^{n} k^2$ such triples.
b) Show that the number of such triples with $0 \leq i < j < k$ and the number of such triples with $0 \leq j < i < k$ both equal $C(n + 1, 3)$.
c) Show that the number of such triples with $0 \leq i = j < k$ equals $C(n + 1, 2)$.
d) Combining part (a) with parts (b) and (c), conclude that

$$\sum_{k=1}^{n} k^2 = 2C(n + 1, 3) + C(n + 1, 2) = n(n + 1)(2n + 1)/6.$$

***33.** How many bit strings of length n, where $n \geq 4$, contain exactly two occurrences of 01?

34. What is the probability that a seven-card poker hand contains

- **a)** four cards of one kind and three cards of a second kind?
- **b)** three cards of one kind and pairs of each of two different kinds?
- **c)** pairs of each of three different kinds and a single card of a fourth kind?
- **d)** pairs of each of two different kinds and three cards of a third, fourth, and fifth kind?
- **e)** cards of seven different kinds?
- **f)** a seven-card flush?
- **g)** a seven-card straight?
- **h)** a seven-card straight flush?

35. What is the probability that a 13-card bridge hand contains

- **a)** all 13 hearts?
- **b)** 13 cards of the same suit?
- **c)** 7 spades and 6 clubs?
- **d)** 7 cards of one suit and 6 cards of a second suit?
- **e)** 4 diamonds, 6 hearts, 2 spades, and 1 club?
- **f)** 4 cards of one suit, 6 cards of a second suit, 2 cards of a third suit, and 1 card of the fourth suit?

36. Suppose that p and q are primes and $n = pq$. What is the probability that a randomly chosen positive integer less than n is not divisible by either p or q?

***37.** Suppose that m and n are positive integers. What is the probability that a randomly chosen positive integer less than mn is not divisible by either m or n?

38. Suppose that $E_1, E_2, \ldots, E_n$ are n events with $p(E_i) > 0$ for $i = 1, 2, \ldots, n$. Show that

$$\begin{aligned} p(E_1 \cap E_2 \cap \cdots \cap E_n) &\\ = p(E_1)p(E_2 \mid E_1)p(E_3 \mid E_1 \cap E_2) &\\ \cdots p(E_n \mid E_1 \cap E_2 \cap \cdots \cap E_{n-1}). & \end{aligned}$$

39. We say that the events $E_1, E_2, \ldots, E_n$ are **mutually independent** if $p(E_{i_1} \cap E_{i_2} \cap \cdots \cap E_{i_m}) = p(E_{i_1})p(E_{i_2}) \cdots p(E_{i_m})$ whenever i_j, $j = 1, 2, \ldots, m$, are integers with $1 \leq i_1 < i_2 < \cdots < i_m \leq n$ and $m \geq 2$.

- **a)** Write out the conditions required for three events E_1, E_2, E_3 to be mutually independent.
- **b)** Let E_1, E_2, and E_3 be the events that the first flip comes up heads, that the second flip comes up tails, and that the third flip comes up tails, respectively, when a fair coin is flipped three times. Are E_1, E_2, and E_3 mutually independent?
- **c)** Let E_1, E_2, and E_3 be the events that the first flip comes up heads, that the third flip comes up heads, and that an even number of heads come up, respectively, when a fair coin is flipped three times. Are E_1, E_2, and E_3 mutually independent?
- **d)** How many conditions must be checked to show that n events are mutually independent?

***40.** Suppose that E and F are events with $p(F) \neq 0$. Show that the probability of E is the weighted average of the probability of E given F and the probability of E given the complement of F, $\overline{F}$, where the weights are the probabilities of F and $\overline{F}$, respectively. That is,

$$p(E) = p(E \mid F)p(F) + p(E \mid \overline{F})p(\overline{F}).$$

[*Hint:* Use the fact that $E = (E \cap F) \cup (E \cap \overline{F})$.]

***41.** Suppose that E is an event from a sample space S and that $F_1, F_2, \ldots, F_n$ are mutually exclusive events such that $\bigcup_{i=1}^{n} F_i = S$. Assume that $p(E) \neq 0$ and $p(F_i) \neq 0$ for $i = 1, 2, \ldots, n$. Show that

$$p(F_j \mid E) = \frac{p(E \mid F_j)p(F_j)}{\sum_{i=1}^{n} p(E \mid F_i)p(F_i)}.$$

[*Hint:* Use the fact that $E = \bigcup_{i=1}^{n}(E \cap F_i)$.] This result is known as **Bayes' formula,** since it was developed by the English philosopher Thomas Bayes.

***42.** A space probe near Neptune communicates with Earth using bit strings. Suppose that in its transmissions it sends a 1 one-third of the time and a 0 two-thirds of the time. When a 0 is sent, the probability it is received correctly is 0.9, and the probability it is received incorrectly (as a 1) is 0.1. When a 1 is sent, the probability it is received correctly is 0.8, and the probability it is received incorrectly (as a 0) is 0.2.

- **a)** Use Exercise 40 to find the probability that a 0 is received.
- **b)** Use Bayes' formula, given in Exercise 41, to find the probability that a 0 was transmitted, given that a 0 was received.

43. Let X be a random variable on a sample space S. Show that $V(aX + b) = a^2V(X)$ whenever a and b are real numbers.

44. Show that if m is a positive integer, then the probability that the mth success occurs on the $(m + n)$th trial when independent Bernoulli trials, each with probability p of success, are run, is $\binom{n+m-1}{n} q^n p^m$.

45. A professor writes 20 multiple-choice questions, each with the possible answer a, b, c, or d, for a discrete mathematics test. If the number of questions with a, b, c, and d as their answer is 8, 3, 4, and 5, respectively, how many different answer keys are possible, if the questions can be placed in any order?

46. How many different arrangements are there of eight people seated at a round table, where two arrangements are considered the same if one can be obtained from the other by a rotation?

47. How many ways are there to assign 24 students to five faculty advisors?

48. How many ways are there to choose a dozen apples from a bushel containing 20 indistinguishable

Delicious apples, 20 indistinguishable Macintosh apples, and 20 indistinguishable Granny Smith apples, if at least three of each kind must be chosen?

49. How many solutions are there to the equation $x_1 + x_2 + x_3 = 17$, where x_1, x_2, and x_3 are nonnegative integers with
  a) $x_1 > 1$, $x_2 > 2$, and $x_3 > 3$?
  b) $x_1 < 6$ and $x_3 > 5$?
  c) $x_1 < 4$, $x_2 < 3$, and $x_3 > 5$?

50. a) How many different strings can be made from the word *PEPPERCORN* when all the letters are used?
  b) How many of these strings start and end with the letter *P*?
  c) In how many of these strings are the three letter *P*s consecutive?

51. How many subsets of a set with 10 elements
  a) have fewer than 5 elements?
  b) have more than 7 elements?
  c) have an odd number of elements?

52. A witness to a hit-and-run accident tells the police that the license plate of the car in the accident, which contains three letters followed by three digits, starts with the letters AS and contains both the digits 1 and 2. How many different license plates can fit this description?

53. How many ways are there to put n identical objects into m distinct containers so that no container is empty?

54. How many ways are there to seat six boys and eight girls in a row of chairs so that no two boys are seated next to each other?

55. Devise an algorithm for generating all the r-permutations of a finite set when repetition is allowed.

56. Devise an algorithm for generating all the r-combinations of a finite set when repetition is allowed.

Computer Projects

WRITE PROGRAMS WITH THE FOLLOWING INPUT AND OUTPUT.

1. Given a positive integer n and a nonnegative integer not exceeding n, find the number of r-permutations and r-combinations of a set with n elements.

2. Given positive integers n and r, find the number of r-permutations when repetition is allowed and r-combinations when repetition is allowed of a set with n elements.

3. Given a positive integer n, find the probability of selecting the six integers from the set $\{1, 2, \ldots, n\}$ that were mechanically selected in a lottery.

4. Given a sequence of positive integers, find the longest increasing and the longest decreasing subsequence of the sequence.

5. Simulate repeated trials of the Monty Hall Three Door problem to calculate the probability of winning with each strategy.

***6.** Given an equation $x_1 + x_2 + \cdots + x_n = C$, where C is a constant, and $x_1, x_2, \ldots, x_n$ are nonnegative integers, list all the solutions.

7. Given a positive integer n, list all the permutations of the set $\{1, 2, 3, \ldots, n\}$ in lexicographic order.

8. Given a positive integer n and a nonnegative integer r not exceeding n, list all the r-combinations of the set $\{1, 2, 3, \ldots, n\}$ in lexicographic order.

9. Given a positive integer n and a nonnegative integer r not exceeding n, list all the r-permutations of the set $\{1, 2, 3, \ldots, n\}$ in lexicographic order.

10. Given a positive integer n, list all the combinations of the set $\{1, 2, 3, \ldots, n\}$.

11. Given positive integers n and r, list all the r-permutations, with repetition allowed, of the set $\{1, 2, 3, \ldots, n\}$.

12. Given positive integers n and r, list all the r-combinations, with repetition allowed, of the set $\{1, 2, 3, \ldots, n\}$.

13. Given a positive integer n, generate a random permutation of the set $\{1, 2, 3, \ldots n\}$. (See Exercise 14 in Section 4.7.)

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. Find the number of possible outcomes in a two-team playoff when the winner is the first team to win 5 out of 9, 6 out of 11, 7 out of 13, and 8 out of 15

2. Which binomial coefficients are odd? Can you formulate a conjecture based on numerical evidence?

3. It is not known whether the binomial coefficient

$C(2n, n)$ must be divisible by the square of a prime or whether the largest exponent in the prime factorization of $C(2n, n)$ grows without bound as n grows. Explore these questions by finding the smallest and largest powers of primes in the factorization of $C(2n, n)$ for as many positive integers n as feasible.

4. Find the probabilities of each type of hand in five-card poker and rank the types of hands by their probability.
5. Find some conditions so that the expected value of buying a \$1 lottery ticket in the New Jersey pick-six lottery has an expected value of more than \$1. To win you have to select the six numbers drawn, where order does not matter, from the positive integers 1 to 48, inclusive. The winnings are split evenly among holders of winning tickets. Be sure to consider the total size of the pot going into the drawing and the number of people buying tickets.
6. Estimate the probability that two integers selected at random are relatively prime by testing a large number of randomly selected pairs of integers. Look up the theorem that gives this probability and compare your results with the correct probability.
7. Determine the number of people needed to ensure that the probability at least two of them have the same day of the year as their birthday is at least 70 percent, at least 80 percent, at least 90 percent, at least 95 percent, at least 98 percent, and at least 99 percent.
8. Generate all the permutations of a set with eight elements.
9. Generate all the 6-permutations of a set with nine elements.
10. Generate all combinations of a set with eight elements.
11. Generate all 5-combinations with repetition allowed of a set with seven elements.
12. Generate a list of 100 randomly selected permutations of the set of the first 100 positive integers. (See Exercise 14 in Section 4.7.)

Writing Projects

RESPOND TO THE FOLLOWING WITH ESSAYS USING OUTSIDE SOURCES.

1. Describe some of the earliest uses of the pigeonhole principle by Dirichlet and other mathematicians.
2. Discuss ways in which the current telephone numbering plan can be extended to accommodate the rapid demand for more telephone numbers. (See if you can find some of the proposals coming from the telecommunications industry.) For each new numbering plan you discuss, show how to find the number of different telephone numbers it supports.
3. Many combinatorial identities are described in this book. Find some sources of such identities and describe important combinatorial identities besides those already introduced in this book. Give some representative proofs, including combinatorial ones, of some of these identities.
4. Describe the origins of probability theory and the first uses of this theory.
5. Describe the different bets you can make when you play roulette. Find the probability of each of these bets in the American version where the wheel contains the numbers 0 and 00. Which is the best bet and which is the worst for you?
6. Discuss the probability of winning when you play the game of blackjack versus a casino. Is there a winning strategy for the person playing against the house?
7. Describe the different models used to model the distribution of particles in statistical mechanics, including Maxwell–Boltzmann, Bose–Einstein, and Fermi–Dirac statistics. In each case, describe the counting techniques used in the model.
8. Define the Stirling numbers of the first kind and describe some of their properties and the identities they satisfy.
9. Define the Stirling numbers of the second kind and describe some of their properties and the identities they satisfy.
10. Define Ramsey numbers, state and prove the theorem of Ramsey that shows they exist, and describe what is currently known about them.
11. Describe additional ways to generate all the permutations of a set with n elements besides those found in Section 4.7. Compare these algorithms and the the algorithms described in the text and exercises of Section 4.7 in terms of their computational complexity.
12. Describe at least one way to generate all the partitions of a positive integer n. (See Exercise 35 in Section 3.3.)

5

Advanced Counting Techniques

Many counting problems cannot be solved easily using the methods discussed in Chapter 4. One such problem is: How many bit strings of length n do not contain two consecutive zeros? To solve this problem, let a_n be the number of such strings of length n. An argument can be given that shows $a_{n+1} = a_n + a_{n-1}$. This equation, called a recurrence relation, and the initial conditions $a_1 = 2$ and $a_2 = 3$ determine the sequence $\{a_n\}$. Moreover, an explicit formula can be found for a_n from the equation relating the terms of the sequence. As we will see, a similar technique can be used to solve many different types of counting problems.

We will also see that many counting problems can be solved using formal power series, called generating functions, where the coefficients of powers of x represent terms of sequence we are interested in. Besides solving counting problems, we will also be able to use generating functions to solve recurrence relations and to prove combinatorial identities.

Many other kinds of counting problems cannot be solved using the techniques discussed in Chapter 4, such as: How many ways are there to assign seven jobs to three employees so that each employee is assigned at least one job? How many primes are there less than 1000? Both of these problems can be solved by counting the number of elements in the union of sets. We will develop a technique, called the principle of inclusion–exclusion, that counts the number of elements in unions of sets, and we will show how this principle can be used to solve counting problems.

The techniques studied in this chapter, together with the basic techniques of Chapter 4, can be used to solve many counting problems.

5.1

Recurrence Relations

INTRODUCTION

The number of bacteria in a colony doubles every hour. If a colony begins with five bacteria, how many will be present in n hours? To solve this problem, let a_n be the number of bacteria at the end of n hours. Since the number of bacteria doubles every hour, the relationship $a_n = 2a_{n-1}$ holds whenever n is a positive integer. This relationship, together with the initial condition $a_0 = 5$, uniquely determines a_n for all nonnegative integers n. We can find a formula for a_n from this information.

Some of the counting problems that cannot be solved using the techniques discussed in Chapter 4 can be solved by finding relationships, called recurrence relations, between the terms of a sequence, as was done in the problem involving bacteria. We will study a variety of counting problems that can be modeled using recurrence relations.

We will develop methods in this section and in the following section for finding explicit formulae for the terms of sequences that satisfy certain types of recurrence relations.

RECURRENCE RELATIONS

In Chapter 3 we discussed how sequences can be defined recursively. Recall that a recursive definition of a sequence specifies one or more initial terms and a rule for determining subsequent terms from those that precede them. Recursive definitions can be used to solve counting problems. When they are, the rule for finding terms from those that precede them is called a **recurrence relation.**

DEFINITION 1. A *recurrence relation* for the sequence $\{a_n\}$ is an equation that expresses a_n in terms of one or more of the previous terms of the sequence, namely, $a_0, a_1, \ldots, a_{n-1}$, for all integers n with $n \geq n_0$, where n_0 is a nonnegative integer. A sequence is called a *solution* of a recurrence relation if its terms satisfy the recurrence relation.

EXAMPLE 1 Let $\{a_n\}$ be a sequence that satisfies the recurrence relation $a_n = a_{n-1} - a_{n-2}$ for $n = 2, 3, 4, \ldots$, and suppose that $a_0 = 3$ and $a_1 = 5$. What are a_2 and a_3?

Solution: We see from the recurrence relation that $a_2 = a_1 - a_0 = 5 - 3 = 2$ and $a_3 = a_2 - a_1 = 2 - 5 = -3$. ■

EXAMPLE 2 Determine whether the sequence $\{a_n\}$ is a solution of the recurrence relation $a_n = 2a_{n-1} - a_{n-2}$ for $n = 2, 3, 4, \ldots$, where $a_n = 3n$ for every nonnegative integer n. Answer the same question where $a_n = 2^n$ and where $a_n = 5$.

Solution: Suppose that $a_n = 3n$ for every nonnegative integer n. Then, for $n \geq 2$, we see that $2a_{n-1} - a_{n-2} = 2[3(n-1)] - 3(n-2) = 3n = a_n$. Therefore, $\{a_n\}$, where $a_n = 3n$, is a solution of the recurrence relation.

Suppose that $a_n = 2^n$ for every nonnegative integer n. Note that $a_0 = 1$, $a_1 = 2$, and $a_2 = 4$. Since $2a_1 - a_0 = 2 \cdot 2 - 1 = 3 \neq a_2$, we see that $\{a_n\}$, where $a_n = 2^n$, is not a solution of the recurrence relation.

Suppose that $a_n = 5$ for every nonnegative integer n. Then for $n \geq 2$, we see that $a_n = 2a_{n-1} - a_{n-2} = 2 \cdot 5 - 5 = 5 = a_n$. Therefore, $\{a_n\}$, where $a_n = 5$, is a solution of the recurrence relation. ■

The **initial conditions** for a sequence specify the terms that precede the first term where the recurrence relation takes effect. For instance, in Example 1, $a_0 = 3$ and $a_1 = 5$ are the initial conditions. The recurrence relation and initial conditions uniquely determine a sequence. This is the case since a recurrence relation, together with initial conditions, provide a recursive definition of the sequence. Any term of the sequence can be found from the initial conditions using the recurrence relation a sufficient number of times. However, there are better ways for computing the terms of certain classes of sequences defined by recurrence relations and initial conditions. We will discuss these methods in this section and in the following section.

MODELING WITH RECURRENCE RELATIONS

We can use recurrence relations to model a wide variety of problems, such as finding compound interest, counting rabbits on an island, determining the number of moves in the Tower of Hanoi puzzle, and counting bit strings with certain properties.

EXAMPLE 3 Compound Interest Suppose that a person deposits \$10,000 in a savings account at a bank yielding 11% per year with interest compounded annually. How much will be in the account after 30 years?

Solution: To solve this problem, let P_n denote the amount in the account after n years. Since the amount after n years equals the amount in the account after $n-1$ years plus interest for the nth year, we see that the sequence $\{P_n\}$ satisfies the recurrence relation

$$P_n = P_{n-1} + 0.11P_{n-1} = (1.11)P_{n-1}.$$

The initial condition is $P_0 = 10{,}000$.

We can use an iterative approach to find a formula for P_n. Note that

$$\begin{aligned}
P_1 &= (1.11)P_0\\
P_2 &= (1.11)P_1 = (1.11)^2P_0\\
P_3 &= (1.11)P_2 = (1.11)^3P_0\\
&\ \ \vdots\\
P_n &= (1.11)P_{n-1} = (1.11)^nP_0.
\end{aligned}$$

When we insert the initial condition $P_0 = 10{,}000$, the formula $P_n = (1.11)^n10{,}000$ is obtained. We can use mathematical induction to establish its validity. That the formula is valid for $n = 0$ is a consequence of the initial condition. Now assume that $P_n = (1.11)^n10{,}000$. Then, from the recurrence relation and the induction hypothesis,

$$P_{n+1} = (1.11)P_n = (1.11)(1.11)^n10{,}000 = (1.11)^{n+1}10{,}000.$$

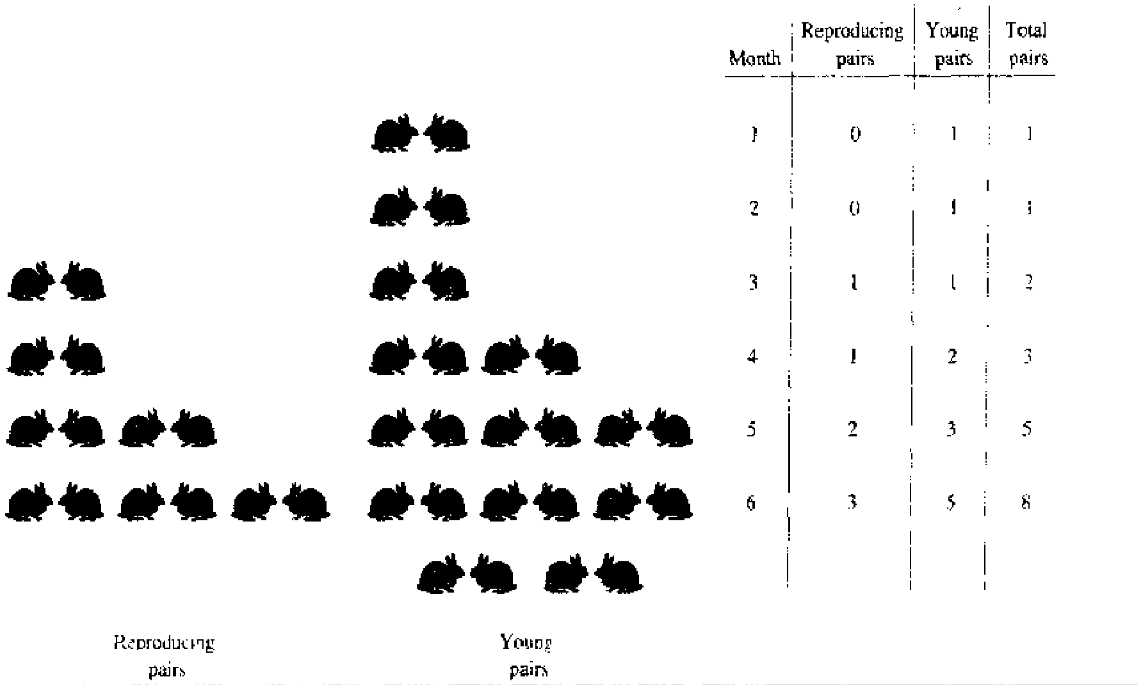
This shows that the explicit formula for P_n is valid.

Inserting $n = 30$ into the formula $P_n = (1.11)^n10{,}000$ shows that after 30 years the account contains

$$P_{30} = (1.11)^{30}10{,}000 = \$228{,}922.97.$$ ■

The next example shows how the population of rabbits on an island can be modeled using a recurrence relation.

EXAMPLE 4 Rabbits and the Fibonacci Numbers Consider the following problem, which was originally posed by Leonardo di Pisa, also known as Fibonacci, in the thirteenth century in his book *Liber abaci*. A young pair of rabbits (one of each sex) is placed on an island. A pair of rabbits does not breed until they are 2 months old. After they are 2 months old, each pair of rabbits produces another pair each month, as shown in Figure 1. Find a recurrence relation for the number of pairs of rabbits on the island after n months, assuming that no rabbits ever die.



Month	Reproducing pairs	Young pairs	Total pairs
1	0	1	1
2	0	1	1
3	1	1	2
4	1	2	3
5	2	3	5
6	3	5	8

FIGURE 1 Rabbits on an Island.

Solution: Denote by f_n the number of pairs of rabbits after n months. We will show that f_n, $n = 1, 2, 3, \ldots$, are the terms of the Fibonacci sequence.

The rabbit population can be modeled using a recurrence relation. At the end of the first month, the number of pairs of rabbits on the island is $f_1 = 1$. Since this pair does not breed during the second month, $f_2 = 1$ also. To find the number of pairs after n months, add the number on the island the previous month, f_{n-1}, and the number of newborn pairs, which equals f_{n-2}, since each newborn pair comes from a pair at least 2 months old.

Consequently, the sequence $\{f_n\}$ satisfies the recurrence relation

$$f_n = f_{n-1} + f_{n-2}$$

for $n \geq 3$ together with the initial conditions $f_1 = 1$ and $f_2 = 1$. Since this recurrence relation and the initial conditions uniquely determine this sequence, the number of pairs of rabbits on the island after n months is given by the nth Fibonacci number. ■

The next example involves a famous puzzle.

EXAMPLE 5

web

The Tower of Hanoi A popular puzzle of the late nineteenth century, called the Tower of Hanoi, consists of three pegs mounted on a board together with disks of different sizes. Initially these disks are placed on the first peg in order of size, with the largest on the bottom (as shown in Figure 2). The rules of the puzzle allow disks to be moved one at a time from one peg to another as long as a disk is never placed on top of a smaller disk. The goal of the puzzle is to have all the disks on the second peg in order of size, with the largest on the bottom.

Let H_n denote the number of moves needed to solve the Tower of Hanoi problem with n disks. Set up a recurrence relation for the sequence $\{H_n\}$.

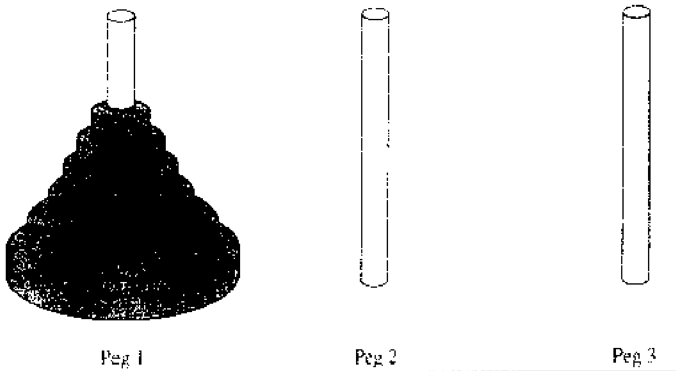


FIGURE 2 **The Initial Position in the Tower of Hanoi.**

Solution: Begin with n disks on peg 1. We can transfer the top $n-1$ disks, following the rules of the puzzle, to peg 3 using H_{n-1} moves (see Figure 3 for an illustration of the pegs and disks at this point). We keep the largest disk fixed during these moves. Then, we use one move to transfer the largest disk to the second peg. We can transfer the $n-1$ disks on peg 3 to peg 2 using H_{n-1} additional moves, placing them on top of the largest disk, which always stays fixed on the bottom of peg 2. Moreover, it is easy to see that the puzzle cannot be solved using fewer steps. This shows that

$$H_n = 2H_{n-1} + 1.$$

The initial condition is $H_1 = 1$, since one disk can be transferred from peg 1 to peg 2, according to the rules of the puzzle, in one move.

We can use an iterative approach to solve this recurrence relation. Note that

$$\begin{aligned} H_n &= 2H_{n-1} + 1 \\ &= 2(2H_{n-2} + 1) + 1 = 2^2H_{n-2} + 2 + 1 \\ &= 2^2(2H_{n-3} + 1) + 2 + 1 = 2^3H_{n-3} + 2^2 + 2 + 1 \\ &\vdots \\ &= 2^{n-1}H_1 + 2^{n-2} + 2^{n-3} + \cdots + 2 + 1 \\ &= 2^{n-1} + 2^{n-2} + \cdots + 2 + 1 \\ &= 2^n - 1. \end{aligned}$$

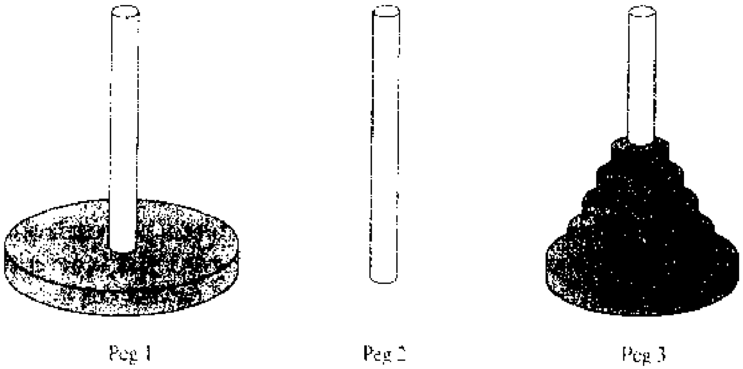


FIGURE 3 **An Intermediate Position in the Tower of Hanoi.**

We have used the recurrence relation repeatedly to express H_n in terms of previous terms of the sequence. In the next to last equality, the initial condition $H_1 = 1$ has been used. The last equality is based on the formula for the sum of the terms of a geometric series, which can be found in Example 5 in Section 3.2.

The iterative approach has produced the solution to the recurrence relation $H_n = 2H_{n-1} + 1$ with the initial condition $H_1 = 1$. This formula can be proved using mathematical induction. This is left as an exercise for the reader at the end of the section.

An ancient myth tells us that there is a tower in Hanoi where monks are transferring 64 gold disks from one peg to another, according to the rules of the puzzle. They take 1 second to move a disk. The myth says that the world will end when they finish the puzzle. How long after the monks started will the world end?

From the explicit formula, the monks require

$$2^{64} - 1 = 18{,}446{,}744{,}073{,}709{,}551{,}615$$

moves to transfer the disks. Making one move per second, it will take them more than 500 billion years to solve the puzzle, so the world should survive a while longer than it already has. ■

web

Remark: Many people have studied variations of the original Tower of Hanoi puzzle discussed in Example 5. Some variations use more pegs, some allow disks to be of the same size, and some restrict the types of allowable disk moves. One of the oldest and most interesting variations is the **Reve's puzzle**,* proposed in 1907 by Henry Dudeney in his book *The Canterbury Puzzles*. The Reve's puzzle involves pilgrims challenged by the Reve to move a stack of cheeses of varying sizes from the first of four stools to another stool without ever placing a cheese on one of smaller diameter. The Reve's puzzle, expressed in terms of pegs and disks, follows the same rules as the Tower of Hanoi puzzle, except that four pegs are used. You may find it surprising that no one has been able to establish the minimum number of moves required to solve this puzzle for n disks. However, there is a conjecture, now more than 50 years old, that the minimum number of moves required equals the number of moves used by an algorithm invented by Frame and Stewart in 1939. (See Exercises 48–55 at the end of this section and [St94] for more information.)

Example 6 illustrates how recurrence relations can be used to count bit strings of a specified length that have a certain property.

EXAMPLE 6 Find a recurrence relation and give initial conditions for the number of bit strings of length n that do not have two consecutive 0s. How many such bit strings are there of length five?

Solution: Let a_n denote the number of bit strings of length n that do not have two consecutive 0s. To obtain a recurrence relation for $\{a_n\}$, note that by the sum rule, the number of bit strings of length n that do not have two consecutive 0s equals the number of such bit strings ending with a 0 plus the number of such bit strings ending with a 1. We will assume that $n \geq 3$, so that the bit string has at least three bits.

**Reve*, more commonly spelled *reeve*, is an archaic word for *governor*.

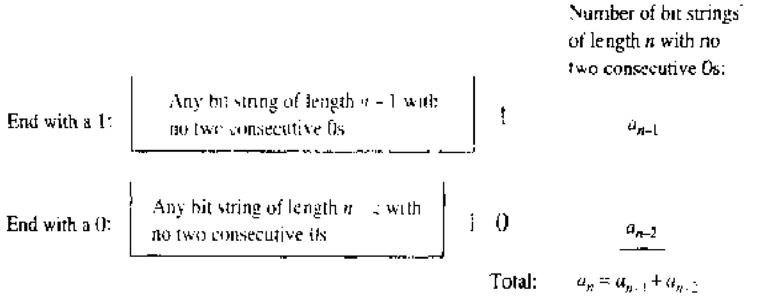

Number of bit strings of length n with no two consecutive 0s:
End with a 1:
Any bit string of length $n-1$ with no two consecutive 0s
1
a_{n-1}
End with a 0:
Any bit string of length $n-2$ with no two consecutive 0s
1 0
a_{n-2}
Total:
$a_n = a_{n-1} + a_{n-2}$

FIGURE 4 Counting Bit Strings of Length n with No Two Consecutive 0s.

The bit strings of length n ending with 1 that do not have two consecutive 0s are precisely the bit strings of length $n-1$ with no two consecutive 0s with a 1 added at the end. Consequently, there are a_{n-1} such bit strings.

Bit strings of length n ending with a 0 that do not have two consecutive 0s must have 1 as their $(n-1)$st bit; otherwise they would end with a pair of 0s. It follows that the bit strings of length n ending with a 0 that have no two consecutive 0s are precisely the bit strings of length $n-2$ with no two consecutive 0s with 10 added at the end. Consequently, there are a_{n-2} such bit strings.

We conclude, as illustrated in Figure 4, that

$$a_n = a_{n-1} + a_{n-2}$$

for $n \geq 3$.

The initial conditions are $a_1 = 2$, since both bit strings of length one, 0 and 1 do not have consecutive 0s, and $a_2 = 3$, since the valid bit strings of length two are 01, 10, and 11. To obtain a_5, we use the recurrence relation three times to find that

$$\begin{aligned} a_3 &= a_2 + a_1 = 3 + 2 = 5, \\ a_4 &= a_3 + a_2 = 5 + 3 = 8, \\ a_5 &= a_4 + a_3 = 8 + 5 = 13. \end{aligned}$$

■

Remark: Note that $\{a_n\}$ satisfies the same recurrence relation as the Fibonacci sequence. Since $a_1 = f_3$ and $a_2 = f_4$ it follows that $a_n = f_{n+2}$.

The next example shows how a recurrence relation can be used to model the number of codewords that are allowable using certain validity checks.

EXAMPLE 7 Codeword Enumeration A computer system considers a string of decimal digits a valid codeword if it contains an even number of 0 digits. For instance, 1230407869 is valid, whereas 120987045608 is not valid. Let a_n be the number of valid n-digit codewords. Find a recurrence relation for a_n.

Solution: Note that $a_1 = 9$ since there are 10 one-digit strings, and only one, namely, the string 0, is not valid. A recurrence relation can be derived for this sequence by considering how a valid n-digit string can be obtained from strings of $n-1$ digits. There are two ways to form a valid string with n digits from a string with one fewer digit.

First, a valid string of n digits can be obtained by appending a valid string of $n-1$ digits with a digit other than 0. This appending can be done in nine ways. Hence, a valid string with n digits can be formed in this manner in $9a_{n-1}$ ways.

Second, a valid string of n digits can be obtained by appending a 0 to a string of length $n-1$ that is not valid. (This produces a string with an even number of 0 digits since the invalid string of length $n-1$ has an odd number of 0 digits.) The number of ways that this can be done equals the number of invalid $(n-1)$-digit strings. Since there are 10^{n-1} strings of length $n-1$, and a_{n-1} are valid, there are $10^{n-1}-a_{n-1}$ valid n-digit strings obtained by appending an invalid string of length $n-1$ with a 0.

Since all valid strings of length n are produced in one of these two ways, it follows that there are

$$\begin{aligned} a_n &= 9a_{n-1} + (10^{n-1} - a_{n-1}) \\ &= 8a_{n-1} + 10^{n-1} \end{aligned}$$

valid strings of length n. ■

The next example establishes a recurrence relation that appears in many different contexts.

EXAMPLE 8 Find a recurrence relation for C_n, the number of ways to parenthesize the product of $n+1$ numbers, $x_0 \cdot x_1 \cdot x_2 \cdot \cdots \cdot x_n$, to specify the order of multiplication. For example, $C_3 = 5$ since there are five ways to parenthesize $x_0 \cdot x_1 \cdot x_2 \cdot x_3$ to determine the order of multiplication: $((x_0 \cdot x_1) \cdot x_2) \cdot x_3$, $(x_0 \cdot (x_1 \cdot x_2)) \cdot x_3$, $(x_0 \cdot x_1) \cdot (x_2 \cdot x_3)$, $x_0 \cdot ((x_1 \cdot x_2) \cdot x_3)$, and $x_0 \cdot (x_1 \cdot (x_2 \cdot x_3))$.

Solution: To develop a recurrence relation for C_n, we note that however we insert parentheses in the product $x_0 \cdot x_1 \cdot x_2 \cdot \cdots \cdot x_n$, one "$\cdot$" operator remains outside all parentheses, namely, the operator for the final multiplication to be performed. [For example, in $(x_0 \cdot (x_1 \cdot x_2)) \cdot x_3$, it is the final "$\cdot$", while in $(x_0 \cdot x_1) \cdot (x_2 \cdot x_3)$ it is the second "$\cdot$".] This final operator appears between two of the $n+1$ numbers, say, x_k and x_{k+1}. There are C_kC_{n-k-1} ways to insert parentheses to determine the order of the $n+1$ numbers to be multiplied when the final operator appears between x_k and x_{k+1}, since there are C_k ways to insert parentheses in the product $x_0 \cdot x_1 \cdot \cdots \cdot x_k$ to determine the order in which these $k+1$ numbers are be be multiplied and C_{n-k-1} ways to insert parentheses in the product $x_{k+1} \cdot x_{k+2} \cdot \cdots \cdot x_n$ to determine the order in which these $n-k$ numbers are to be multiplied. Since this final operator can appear between any two of the $n+1$ numbers, it follows that

$$\begin{aligned} C_n &= C_0C_{n-1} + C_1C_{n-2} + \cdots + C_{n-2}C_1 + C_{n-1}C_0 \\ &= \sum_{k=0}^{n-1} C_kC_{n-k-1}. \end{aligned}$$

Note that the initial conditions are $C_0 = 1$ and $C_1 = 1$. This recurrence relation can be solved using the method of generating functions, which will be discussed in Section 5.4. It can be shown that $C_n = C(2n, n)/(n+1)$. (See Exercise 41 at the end of that section.) ■

web The sequence $\{C_n\}$ is the sequence of **Catalan numbers**. This sequence appears as the solution of many different counting problems besides the one considered here (see the chapter on Catalan numbers in [MiRo90] or [Rob84] for details).

Exercises

1. Find the first five terms of the sequence defined by each of the following recurrence relations and initial conditions.
   a) $a_n = 6a_{n-1}, a_0 = 2$
   b) $a_n = a_{n-1}^2, a_1 = 2$
   c) $a_n = a_{n-1} + 3a_{n-2}, a_0 = 1, a_1 = 2$
   d) $a_n = na_{n-1} + n^2a_{n-2}, a_0 = 1, a_1 = 1$
   e) $a_n = a_{n-1} + a_{n-3}, a_0 = 1, a_1 = 2, a_2 = 0$
2. Show that the sequence $\{a_n\}$ is a solution of the recurrence relation $a_n = -3a_{n-1} + 4a_{n-2}$ if
   a) $a_n = 0$.
   b) $a_n = 1$.
   c) $a_n = (-4)^n$.
   d) $a_n = 2(-4)^n + 3$.
3. Is the sequence $\{a_n\}$ a solution of the recurrence relation $a_n = 8a_{n-1} - 16a_{n-2}$ if
   a) $a_n = 0$?
   b) $a_n = 1$?
   c) $a_n = 2^n$?
   d) $a_n = 4^n$?
   e) $a_n = n4^n$?
   f) $a_n = 2 \cdot 4^n + 3n4^n$?
   g) $a_n = (-4)^n$?
   h) $a_n = n^24^n$?
4. For each of the following sequences find a recurrence relation satisfied by this sequence. (The answers are not unique since there are infinitely many different recurrence relations satisfied by any sequence.)
   a) $a_n = 3$
   b) $a_n = 2n$
   c) $a_n = 2n + 3$
   d) $a_n = 5^n$
   e) $a_n = n^2$
   f) $a_n = n^2 + n$
   g) $a_n = n + (-1)^n$
   h) $a_n = n!$
5. Find the solution to each of the following recurrence relations and initial conditions. Use an iterative approach such as that used in Example 5.
   a) $a_n = 3a_{n-1}, a_0 = 2$
   b) $a_n = a_{n-1} + 2, a_0 = 3$
   c) $a_n = a_{n-1} + n, a_0 = 1$
   d) $a_n = a_{n-1} + 2n + 3, a_0 = 4$
   e) $a_n = 2a_{n-1} - 1, a_0 = 1$
   f) $a_n = 3a_{n-1} + 1, a_0 = 1$
   g) $a_n = na_{n-1}, a_0 = 5$
   h) $a_n = 2na_{n-1}, a_0 = 1$
6. A person deposits \$1000 in an account that yields 9% interest compounded yearly.
   a) Set up a recurrence relation for the amount in the account at the end of n years.
   b) Find an explicit formula for the amount in the account at the end of n years.
   c) How much money will the account contain after 100 years?
7. Suppose that the number of bacteria in a colony triples every hour.
   a) Set up a recurrence relation for the number of bacteria after n hours have elapsed.
   b) If 100 bacteria are used to begin a new colony, how many bacteria will be in the colony in 10 hours?
8. Assume that the population of the world in 1999 is 6 billion and is growing at the rate of 1.3% a year.
   a) Set up a recurrence relation for the population of the world n years after 1999.
   b) Find an explicit formula for the population of the world n years after 1999.
   c) What will the population of the world be in 2020?
9. A factory makes custom sports cars at an increasing rate. In the first month only one car is made, in the second month two cars are made, and so on, with n cars made in the nth month.
   a) Set up a recurrence relation for the number of cars produced in the first n months by this factory.
   b) How many cars are produced in the first year?
   c) Find an explicit formula for the number of cars produced in the first n months by this factory.
10. An employee joined a company in 1987 with a starting salary of \$50,000. Every year this employee receives a raise of \$1000 plus 5% of the salary of the previous year.
    a) Set up a recurrence relation for the salary of this employee n years after 1987.
    b) What is the salary of this employee in 1995?
    c) Find an explicit formula for the salary of this employee n years after 1987.
11. Use mathematical induction to verify the formula derived in Example 5 for the number of moves required to complete the Tower of Hanoi puzzle.
12. a) Find a recurrence relation for the number of permutations of a set with n elements.
    b) Use this recurrence relation to find the number of permutations of a set with n elements using iteration.
13. A vending machine dispensing books of stamps accepts only Susan B. Anthony dollar coins, \$1 bills, and \$5 bills.
    a) Find a recurrence relation for the number of ways to deposit n dollars in the vending machine, where the order in which the coins and bills are deposited matters.
    b) What are the initial conditions?

c) How many ways are there to deposit \$10 for a book of stamps?

14. A country uses as currency coins with values of 1 peso, 2 pesos, 5 pesos, and 10 pesos and bills with values of 5 pesos, 10 pesos, 20 pesos, 50 pesos, and 100 pesos. Find a recurrence relation for the number of ways to pay a bill of n pesos if the order in which the coins and bills are paid matters.

15. How many ways are there to pay a bill of 17 pesos using the currency described in Exercise 14, where the order in which coins and bills are paid matters?

*16. a) Find a recurrence relation for the number of strictly increasing sequences of positive integers that have 1 as their first term and n as their last term where n is a positive integer. That is, sequences $a_1, a_2, \ldots, a_k$ where $a_1 = 1$, $a_k = n$, and $a_j < a_{j+1}$ for $j = 1, 2, \ldots, k - 1$.
   b) What are the initial conditions?
   c) How many sequences of the type described in (a) are there when n is a positive integer with $n \geq 2$?

17. a) Find a recurrence relation for the number of bit strings of length n that contain a pair of consecutive 0s.
   b) What are the initial conditions?
   c) How many bit strings of length seven contain two consecutive 0s?

18. a) Find a recurrence relation for the number of bit strings of length n that contain three consecutive 0s.
   b) What are the initial conditions?
   c) How many bit strings of length seven contain three consecutive 0s?

19. a) Find a recurrence relation for the number of bit strings of length n that do not contain three consecutive 0s.
   b) What are the initial conditions?
   c) How many bit strings of length seven do not contain three consecutive 0s?

*20. a) Find a recurrence relation for the number of bit strings that contain the string 01.
   b) What are the initial conditions?
   c) How many bit strings of length seven contain the string 01?

21. a) Find a recurrence relation for the number of ways to climb n stairs if the person climbing the stairs can take one stair or two stairs at a time.
   b) What are the initial conditions?
   c) How many ways can this person climb a flight of eight stairs?

22. a) Find a recurrence relation for the number of ways to climb n stairs if the person climbing the stairs can take one, two, or three stairs at a time.
   b) What are the initial conditions?
   c) How many ways can this person climb a flight of eight stairs?

A string that contains only 0s, 1s, and 2s is called a **ternary string.**

23. a) Find a recurrence relation for the number of ternary strings that do not contain two consecutive 0s.
   b) What are the initial conditions?
   c) How many ternary strings of length six do not contain two consecutive 0s?

24. a) Find a recurrence relation for the number of ternary strings that contain two consecutive 0s.
   b) What are the initial conditions?
   c) How many ternary strings of length six contain two consecutive 0s?

*25. a) Find a recurrence relation for the number of ternary strings that do not contain two consecutive 0s or two consecutive 1s.
   b) What are the initial conditions?
   c) How many ternary strings of length six do not contain two consecutive 0s or two consecutive 1s?

*26. a) Find a recurrence relation for the number of ternary strings that contain either two consecutive 0s or two consecutive 1s.
   b) What are the initial conditions?
   c) How many ternary strings of length six contain two consecutive 0s or two consecutive 1s?

*27. a) Find a recurrence relation for the number of ternary strings that do not contain consecutive symbols which are the same.
   b) What are the initial conditions?
   c) How many ternary strings of length six do not contain consecutive symbols which are the same?

**28. a) Find a recurrence relation for the number of ternary strings that contain two consecutive symbols which are the same.
   b) What are the initial conditions?
   c) How many ternary strings of length six contain consecutive symbols that are the same?

29. Messages are transmitted over a communications channel using two signals. The transmittal of one signal requires 1 microsecond, and the transmittal of the other signal requires 2 microseconds.
   a) Find a recurrence relation for the number of different messages consisting of sequences of these two signals, where each signal in the message is immediately followed by the next signal, that can be sent in n microseconds.
   b) What are the initial conditions?
   c) How many different messages can be sent in 10 microseconds using these two signals?

30. A bus driver pays all tolls, using only nickels and dimes, by throwing one coin at a time into the mechanical toll collector.
   a) Find a recurrence relation for the number of different ways the bus driver can pay a toll of n cents

(where the order in which the coins are used matters).

b) In how many different ways can the driver pay a toll of 45 cents?

31. **a)** Find the recurrence relation satisfied by R_n, where R_n is the number of regions that a plane is divided into by n lines, if no two of the lines are parallel and no three of the lines go through the same point.

b) Find R_n using iteration.

***32.** **a)** Find the recurrence relation satisfied by R_n, where R_n is the number of regions into which the surface of a sphere is divided into by n great circles (which are the intersections of the sphere and planes passing through the center of the sphere), if no three of the great circles go through the same point.

b) Find R_n using iteration.

***33.** **a)** Find the recurrence relation satisfied by S_n, where S_n is the number of regions into which three-dimensional space is divided into by n planes if every three of the planes meet in one point, but no four of the planes go through the same point.

b) Find S_n using iteration.

34. Find a recurrence relation for the number of bit sequences of length n with an even number of 0s.

35. How many bit sequences of length seven contain an even number of 0s?

36. **a)** Find a recurrence relation for the number of ways to completely cover a $2 \times n$ chessboard with 1×2 dominos. (*Hint:* Consider separately the coverings where the position in the top right corner of the chessboard is covered by a domino positioned horizontally and where it is covered by a domino positioned vertically.)

b) What are the initial conditions for the recurrence relation in part (a)?

c) How many ways are there to completely cover a 2×17 chessboard with 1×2 dominos?

37. **a)** Find a recurrence relation for the number of ways to lay out a walkway with slate tiles if the tiles are red, green, or gray, so that no two red tiles are adjacent and tiles of the same color are considered indistinguishable.

b) What are the initial conditions for the recurrence relation in part (a)?

c) How many ways are there to lay out a path of seven tiles as described in part (a)?

38. Show that the Fibonacci numbers satisfy the recurrence relation $f_n = 5f_{n-4} + 3f_{n-5}$ for $n = 5, 6, 7, \ldots$, together with the initial conditions $f_0 = 0$, $f_1 = 1$, $f_2 = 1$, $f_3 = 2$, and $f_4 = 3$. Use this recurrence relation to show that f_{5n} is divisible by 5, for $n = 1, 2, 3, \ldots$.

***39.** Let $S(m, n)$ denote the number of onto functions from a set with m elements to a set with n elements. Show that $S(m, n)$ satisfies the recurrence relation

$$S(m, n) = n^m - \sum_{k=1}^{n-1} C(n, k) S(m, k)$$

whenever $m \geq n$ and $n > 1$, with the initial condition $S(m, 1) = 1$.

40. **a)** Write out all the ways the product $x_0 \cdot x_1 \cdot x_2 \cdot x_3 \cdot x_4$ can be parenthesized to determine the order of multiplication.

b) Use the recurrence relation developed in Example 8 to calculate C_4, the number of ways to parenthesize the product of five numbers so as to determine the order of multiplication. Verify that you listed the correct number of ways in part (a).

c) Check your result in part (b) by finding C_4, using the closed formula for C_n mentioned in the solution of Example 8.

41. **a)** Use the recurrence relation developed in Example 8 to determine C_5, the number of ways to parenthesize the product of six numbers so as to determine the order of multiplication.

b) Check your result with the closed formula for C_5 mentioned in the solution of Example 8.

***42.** In the Tower of Hanoi puzzle, suppose our goal is to transfer all n disks from peg 1 to peg 3, but we cannot move a disk directly between pegs 1 and 3. Each move of a disk must be a move involving peg 2. As usual, we cannot place a disk on top of a smaller disk.

a) Find a recurrence relation for the number of moves required to solve the puzzle for n disks with this added restriction.

b) Solve this recurrence relation to find a formula for the number of moves required to solve the puzzle for n disks.

c) How many different arrangements are there of the n disks on three pegs so that no disk is on top of a smaller disk?

d) Show that every allowable arrangement of the n disks occurs in the solution of this variation of the puzzle.

Exercises 43–47 deal with a variation of the **Josephus problem** described by Graham, Knuth, and Patashnik in [GrKnPa94]. This problem is based on an account by the historian Flauvius Josephus, who was part of a band of 41 Jewish rebels trapped in a cave by the Romans during the Jewish-Roman war of the first century. The rebels preferred suicide to capture; they decided to form a circle and to repeatedly count off around the circle, killing every third rebel left alive. However, Josephus and another rebel did not want to be killed this way; they determined the positions where they should stand to be the last two rebels remaining alive. The variation we consider begins with n people, numbered 1 to n, standing around a circle. In each stage, every second person still left alive is eliminated until only one survives. We denote the number of the survivor by $J(n)$.

43. Determine the value of $J(n)$ for each integer n with $1 \le n \le 16$.

44. Use the values you found in Exercise 43 to conjecture a formula for $J(n)$. (*Hint:* Write $n = 2^m + k$, where m is a nonnegative integer and k is a nonnegative integer less than 2^m.)

45. Show that $J(n)$ satisfies the recurrence relation $J(2n) = 2J(n) - 1$ and $J(2n+1) = 2J(n) + 1$, for $n \ge 1$, and $J(1) = 1$.

46. Use mathematical induction to prove the formula you conjectured in Exercise 44, making use of the recurrence relation from Exercise 45.

47. Determine $J(100)$, $J(1000)$, and $J(10000)$ from your formula for $J(n)$.

Exercises 48–55 involve the Reve's puzzle, the variation of the Tower of Hanoi puzzle with four pegs and n disks. Before presenting these exercises, we describe the Frame–Stewart algorithm for moving the disks from peg 1 to peg 4 so that no disk is ever on top of a smaller one. This algorithm, given the number of disks n as input, depends on a choice of an integer k with $1 \le k \le n$. When there is only one disk, move it from peg 1 to peg 4 and stop. For $n > 1$, the algorithm proceeds recursively, using the following three steps. Recursively move the stack of the $n - k$ smallest disks from peg 1 to peg 2, using all four pegs. Next move the stack of the k largest disks from peg 1 to peg 4, using the three-peg algorithm from the Tower of Hanoi puzzle without using the peg holding the $n - k$ smallest disks. Finally, recursively move the smallest $n - k$ disks to peg 4, using all four pegs. Frame and Stewart showed that to produce the fewest moves using their algorithm, k should be chosen to be the smallest integer such that n does not exceed $t_k = k(k+1)/2$, the kth triangular number, i.e., $t_{k-1} < n \le t_k$. The unsettled conjecture, known as **Frame's conjecture**, is that this algorithm uses the fewest number of moves required to solve the puzzle, no matter how the disks are moved.

48. Show that the Reve's puzzle with three disks can be solved using five, and no fewer, moves.

49. Show that the Reve's puzzle with four disks can be solved using nine, and no fewer, moves.

50. Describe the moves made by the Frame–Stewart algorithm, with k chosen so that the fewest moves are required, for

a) 5 disks. **b)** 6 disks. **c)** 7 disks. **d)** 8 disks.

***51.** Show that if $R(n)$ is the number of moves used by the Frame–Stewart algorithm to solve the Reve's puzzle with n disks, where k is chosen to be the smallest integer with $n \le k(k+1)/2$, then $R(n)$ satisfies the recurrence relation with $R(n) = 2R(n-k) + 2^k - 1$, with $R(0) = 0$ and $R(1) = 1$.

***52.** Show that if k is as chosen in Exercise 51, then $R(n) - R(n-1) = 2^{k-1}$.

***53.** Show that if k is as chosen in Exercise 51, then $R(n) = \sum_{i=1}^{k} i2^{i-1} - (t_k - n)2^{k-1}$.

***54.** Use Exercise 53 to give an upper bound on the number of moves required to solve the Reve's puzzle for all integers n with $1 \le n \le 25$.

***55.** Show that $R(n)$ is $O(\sqrt{n}2^{\sqrt{2n}})$

Let $\{a_n\}$ be a sequence of real numbers. The **backward differences** of this sequence are defined recursively as follows. The **first difference** ∇a_n is

$$\nabla a_n = a_n - a_{n-1}.$$

The **$(k+1)$th difference** $\nabla^{k+1} a_n$ is obtained from $\nabla^k a_n$ by

$$\nabla^{k+1} a_n = \nabla^k a_n - \nabla^k a_{n-1}.$$

56. Find ∇a_n for the sequence $\{a_n\}$ where

a) $a_n = 4$. **b)** $a_n = 2n$. **c)** $a_n = n^2$. **d)** $a_n = 2^n$.

57. Find $\nabla^2 a_n$ for the sequences in Exercise 34.

58. Show that $a_{n-1} = a_n - \nabla a_n$.

59. Show that $a_{n-2} = a_n - 2\nabla a_n + \nabla^2 a_n$.

***60.** Prove that a_{n-k} can be expressed in terms of $a_n, \nabla a_n, \nabla^2 a_n, \ldots, \nabla^k a_n$.

61. Express the recurrence relation $a_n = a_{n-1} + a_{n-2}$ in terms of a_n, ∇a_n, and $\nabla^2 a_n$.

62. Show that any recurrence relation for the sequence $\{a_n\}$ can be written in terms of $a_n, \nabla a_n, \nabla^2 a_n, \ldots$. The resulting equation involving the sequences and its differences is called a **difference equation.**

5.2

Solving Recurrence Relations

INTRODUCTION

A wide variety of recurrence relations occur in models. Some of these recurrence relations can be solved using iteration or some other ad hoc technique. However, one important class of recurrence relations can be explicitly solved in a systematic way. These are recurrence relations that express the terms of a sequence as linear combinations of previous terms.

DEFINITION 1. A ***linear homogeneous recurrence relation of degree k with constant coefficients*** **is a recurrence relation of the form**

$$a_n = c_1a_{n-1} + c_2a_{n-2} + \cdots + c_ka_{n-k}$$

where $c_1, c_2, \ldots, c_k$ are real numbers, and $c_k \neq 0$.

The recurrence relation in the definition is **linear** since the right-hand side is a sum of multiples of the previous terms of the sequence. The recurrence relation is **homogeneous** since no terms occur that are not multiples of the a_js. The coefficients of the terms of the sequence are all **constants,** rather than functions that depend on n. The **degree** is k because a_n is expressed in terms of the previous k terms of the sequence.

A consequence of the second principle of mathematical induction is that a sequence satisfying the recurrence relation in the definition is uniquely determined by this recurrence relation and the k initial conditions

$$a_0 = C_0, a_1 = C_1, \ldots, a_{k-1} = C_{k-1}.$$

EXAMPLE 1 The recurrence relation $P_n = (1.11)P_{n-1}$ is a linear homogeneous recurrence relation of degree one. The recurrence relation $f_n = f_{n-1} + f_{n-2}$ is a linear homogeneous recurrence relation of degree two. The recurrence relation $a_n = a_{n-5}$ is a linear homogeneous recurrence relation of degree five. ■

Some examples of recurrence relations that are not linear homogeneous recurrence relations with constant coefficients follow.

EXAMPLE 2 The recurrence relation $a_n = a_{n-1} + a_{n-2}^2$ is not linear. The recurrence relation $H_n = 2H_{n-1} + 1$ is not homogeneous. The recurrence relation $B_n = nB_{n-1}$ does not have constant coefficients. ■

Linear homogeneous recurrence relations are studied for two reasons. First, they often occur in modeling of problems. Second, they can be systematically solved.

SOLVING LINEAR HOMOGENEOUS RECURRENCE RELATIONS WITH CONSTANT COEFFICIENTS

The basic approach for solving linear homogeneous recurrence relations is to look for solutions of the form $a_n = r^n$, where r is a constant. Note that $a_n = r^n$ is a solution of the recurrence relation $a_n = c_1a_{n-1} + c_2a_{n-2} + \cdots + c_ka_{n-k}$ if and only if

$$r^n = c_1r^{n-1} + c_2r^{n-2} + \cdots + c_kr^{n-k}.$$

When both sides of this equation are divided by r^{n-k} and the right-hand side is subtracted from the left, we obtain the equivalent equation

$$r^k - c_1r^{k-1} - c_2r^{k-2} - \cdots - c_{k-1}r - c_k = 0.$$

Consequently, the sequence $\{a_n\}$ with $a_n = r^n$ is a solution if and only if r is a solution of this last equation, which is called the **characteristic equation** of the recurrence relation. The solutions of this equation are called the **characteristic roots** of the recurrence relation. As we will see, these characteristic roots can be used to give an explicit formula for all the solutions of the recurrence relation.

We will first develop results that deal with linear homogeneous recurrence relations with constant coefficients of degree two. Then corresponding general results when the degree may be greater than two will be stated. Because the proofs needed to establish the results in the general case are more complicated, they will not be given in the text.

We now turn our attention to linear homogeneous recurrence relations of degree two. First, consider the case when there are two distinct characteristic roots.

THEOREM 1 **Let c_1 and c_2 be real numbers. Suppose that $r^2 - c_1r - c_2 = 0$ has two distinct roots r_1 and r_2. Then the sequence $\{a_n\}$ is a solution of the recurrence relation $a_n = c_1a_{n-1} + c_2a_{n-2}$ if and only if $a_n = \alpha_1r_1^n + \alpha_2r_2^n$ for $n = 0, 1, 2, \ldots$, where α_1 and α_2 are constants.**

Proof: We must do two things to prove the theorem. First, it must be shown that if r_1 and r_2 are the roots of the characteristic equation, and α_1 and α_2 are constants, then the sequence $\{a_n\}$ with $a_n = \alpha_1r_1^n + \alpha_2r_2^n$ is a solution of the recurrence relation. Second, it must be shown that if the sequence $\{a_n\}$ is a solution, then $a_n = \alpha_1r_1^n + \alpha_2r_2^n$ for some constants α_1 and α_2.

Now we will show that if $a_n = \alpha_1r_1^n + \alpha_2r_2^n$, then the sequence $\{a_n\}$ is a solution of the recurrence relation. Since r_1 and r_2 are roots of $r^2 - c_1r - c_2 = 0$, it follows that $r_1^2 = c_1r_1 + c_2$, $r_2^2 = c_1r_2 + c_2$.

From these equations, we see that

$$\begin{aligned} c_1a_{n-1} + c_2a_{n-2} &= c_1(\alpha_1r_1^{n-1} + \alpha_2r_2^{n-1}) + c_2(\alpha_1r_1^{n-2} + \alpha_2r_2^{n-2}) \\ &= \alpha_1r_1^{n-2}(c_1r_1 + c_2) + \alpha_2r_2^{n-2}(c_1r_2 + c_2) \\ &= \alpha_1r_1^{n-2}r_1^2 + \alpha_2r_2^{n-2}r_2^2 \\ &= \alpha_1r_1^n + \alpha_2r_2^n \\ &= a_n. \end{aligned}$$

This shows that the sequence $\{a_n\}$ with $a_n = \alpha_1r_1^n + \alpha_2r_2^n$ is a solution of the recurrence relation.

To show that every solution $\{a_n\}$ of the recurrence relation $a_n = c_1a_{n-1} + c_2a_{n-2}$ has $a_n = \alpha_1r_1^n + \alpha_2r_2^n$ for $n = 0, 1, 2, \ldots$, for some constants α_1 and α_2, suppose that $\{a_n\}$ is a solution of the recurrence relation, and the initial conditions $a_0 = C_0$ and $a_1 = C_1$ hold. It will be shown that there are constants α_1 and α_2 so that the sequence $\{a_n\}$ with $a_n = \alpha_1r_1^n + \alpha_2r_2^n$ satisfies these same initial conditions. This requires that

$$\begin{aligned} a_0 &= C_0 = \alpha_1 + \alpha_2, \\ a_1 &= C_1 = \alpha_1r_1 + \alpha_2r_2. \end{aligned}$$

We can solve these two equations for α_1 and α_2. From the first equation it follows that $\alpha_2 = C_0 - \alpha_1$. Inserting this expression into the second equation gives

$$C_1 = \alpha_1r_1 + (C_0 - \alpha_1)r_2.$$

Hence,

$$C_1 = \alpha_1(r_1 - r_2) + C_0r_2.$$

This shows that

$$\alpha_1 = \frac{(C_1 - C_0r_2)}{r_1 - r_2}$$

and

$$\alpha_2 = C_0 - \alpha_1 = C_0 - \frac{(C_1 - C_0r_2)}{r_1 - r_2} = \frac{C_0r_1 - C_1}{r_1 - r_2},$$

where these expressions for α_1 and α_2 depend on the fact that $r_1 \neq r_2$. (When $r_1 = r_2$, this theorem is not true.) Hence, with these values for α_1 and α_2, the sequence $\{a_n\}$ with $\alpha_1r_1^n + \alpha_2r_2^n$ satisfies the two initial conditions. Since this recurrence relation and these initial conditions uniquely determine the sequence, it follows that $a_n = \alpha_1r_1^n + \alpha_2r_2^n$. □

The characteristic roots of a linear homogeneous recurrence relation with constant coefficients may be complex numbers. Theorem 1 (and also subsequent theorems in this section) still applies in this case. Recurrence relations with complex characteristic roots will not be discussed in the text. Readers familiar with complex numbers may wish to solve Exercises 38 and 39 at the end of this section.

The following examples illustrate the usefulness of the explicit formula given in Theorem 1.

EXAMPLE 3 What is the solution of the recurrence relation

$$a_n = a_{n-1} + 2a_{n-2}$$

with $a_0 = 2$ and $a_1 = 7$?

Solution: Theorem 1 can be used to solve this problem. The characteristic equation of the recurrence relation is $r^2 - r - 2 = 0$. Its roots are $r = 2$ and $r = -1$. Hence, the sequence $\{a_n\}$ is a solution to the recurrence relation if and only if

$$a_n = \alpha_1 2^n + \alpha_2(-1)^n,$$

for some constants α_1 and α_2. From the initial conditions, it follows that

$$a_0 = 2 = \alpha_1 + \alpha_2,$$
$$a_1 = 7 = \alpha_1 \cdot 2 + \alpha_2 \cdot (-1).$$

Solving these two equations shows that $\alpha_1 = 3$ and $\alpha_2 = -1$. Hence, the solution to the recurrence relation and initial conditions is the sequence $\{a_n\}$ with

$$a_n = 3 \cdot 2^n - (-1)^n.$$ ■

EXAMPLE 4 Find an explicit formula for the Fibonacci numbers.

Solution: Recall that the sequence of Fibonacci numbers satisfies the recurrence relation $f_n = f_{n-1} + f_{n-2}$ and also satisfies the initial conditions $f_0 = 0$ and $f_1 = 1$. The roots of the characteristic equation $r^2 - r - 1 = 0$ are $r_1 = (1 + \sqrt{5})/2$ and $r_2 = (1 - \sqrt{5})/2$. Therefore, from Theorem 1 it follows that the Fibonacci numbers are given by

$$f_n = \alpha_1\left(\frac{1+\sqrt{5}}{2}\right)^n + \alpha_2\left(\frac{1-\sqrt{5}}{2}\right)^n,$$

for some constants α_1 and α_2. The initial conditions $f_0 = 0$ and $f_1 = 1$ can be used to find these constants. We have

$$f_0 = \alpha_1 + \alpha_2 = 0,$$
$$f_1 = \alpha_1\left(\frac{1+\sqrt{5}}{2}\right) + \alpha_2\left(\frac{1-\sqrt{5}}{2}\right) = 1.$$

The solution to these simultaneous equations for α_1 and α_2 is

$$\alpha_1 = 1/\sqrt{5}, \qquad \alpha_2 = -1/\sqrt{5}.$$

Consequently, the Fibonacci numbers are given by

$$f_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n.$$

■

Theorem 1 does not apply when there is one characteristic root of multiplicity two. This case can be handled using the following theorem.

THEOREM 2 **Let c_1 and c_2 be real numbers with $c_2 \neq 0$. Suppose that $r^2 - c_1r - c_2 = 0$ has only one root r_0. A sequence $\{a_n\}$ is a solution of the recurrence relation $a_n = c_1a_{n-1} + c_2a_{n-2}$ if and only if $a_n = \alpha_1 r_0^n + \alpha_2 n r_0^n$, for $n = 0, 1, 2, \ldots$, where α_1 and α_2 are constants.**

The proof of Theorem 2 is left as an exercise at the end of the section. The following example illustrates the use of this theorem.

EXAMPLE 5 What is the solution of the recurrence relation

$$a_n = 6a_{n-1} - 9a_{n-2}$$

with initial conditions $a_0 = 1$ and $a_1 = 6$?

Solution: The only root of $r^2 - 6r + 9 = 0$ is $r = 3$. Hence, the solution to this recurrence relation is

$$a_n = \alpha_1 3^n + \alpha_2 n 3^n$$

for some constants α_1 and α_2. Using the initial conditions, it follows that

$$a_0 = 1 = \alpha_1,$$
$$a_1 = 6 = \alpha_1 \cdot 3 + \alpha_2 \cdot 3.$$

Solving these two equations shows that $\alpha_1 = 1$ and $\alpha_2 = 1$. Consequently, the solution to this recurrence relation and the initial conditions is

$$a_n = 3^n + n3^n.$$

■

We will now state the general result about the solution of linear homogeneous recurrence relations with constant coefficients, where the degree may be greater than two, under the assumption that the characteristic equation has distinct roots. The proof of this result will be left as an exercise for the reader.

THEOREM 3 **Let $c_1, c_2, \ldots, c_k$ be real numbers. Suppose that the characteristic equation**

$$r^k - c_1 r^{k-1} - \cdots - c_k = 0$$

has k distinct roots $r_1, r_2, \ldots, r_k$. Then a sequence $\{a_n\}$ is a solution of the recurrence relation

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k}$$

if and only if

$$a_n = \alpha_1 r_1^n + \alpha_2 r_2^n + \cdots + \alpha_k r_k^n$$

for $n = 0, 1, 2, \ldots$, where $\alpha_1, \alpha_2, \ldots, \alpha_k$ are constants.

We illustrate the use of the theorem with an example.

EXAMPLE 6 Find the solution to the recurrence relation

$$a_n = 6a_{n-1} - 11a_{n-2} + 6a_{n-3}$$

with the initial conditions $a_0 = 2$, $a_1 = 5$, and $a_2 = 15$.

Solution: The characteristic polynomial of this recurrence relation is

$$r^3 - 6r^2 + 11r - 6.$$

The characteristic roots are $r = 1$, $r = 2$, and $r = 3$, since $r^3 - 6r^2 + 11r - 6 = (r-1)(r-2)(r-3)$. Hence, the solutions to this recurrence relation are of the form

$$a_n = \alpha_1 \cdot 1^n + \alpha_2 \cdot 2^n + \alpha_3 \cdot 3^n.$$

To find the constants α_1, α_2, and α_3, use the initial conditions. This gives

$$\begin{aligned} a_0 &= 2 = \alpha_1 + \alpha_2 + \alpha_3, \\ a_1 &= 5 = \alpha_1 + \alpha_2 \cdot 2 + \alpha_3 \cdot 3, \\ a_2 &= 15 = \alpha_1 + \alpha_2 \cdot 4 + \alpha_3 \cdot 9. \end{aligned}$$

When these three simultaneous equations are solved for α_1, α_2, and α_3, we find that $\alpha_1 = 1$, $\alpha_2 = -1$, and $\alpha_3 = 2$. Hence, the unique solution to this recurrence relation and the given initial conditions is the sequence $\{a_n\}$ with

$$a_n = 1 - 2^n + 2 \cdot 3^n.$$ ■

We now state the most general result about linear homogeneous recurrence relations with constant coefficients, allowing the characteristic equation to have multiple roots. The key point is that for each root r of the characteristic equation, the general solution has a summand of the form $P(n)r^n$, where $P(n)$ is a polynomial of degree $m - 1$, with m the multiplicity of this root. We leave the proof of this result as a challenging exercise for the reader.

THEOREM 4 **Let $c_1, c_2, \ldots, c_k$ be real numbers. Suppose that the characteristic equation**

$$r^k - c_1 r^{k-1} - \cdots - c_k = 0$$

has t distinct roots $r_1, r_2, \ldots, r_t$ with multiplicities $m_1, m_2, \ldots, m_t$, respectively, so that $m_i \geq 1$ for $i = 1, 2, \ldots, t$ and $m_1 + m_2 + \cdots + m_t = k$. Then a sequence $\{a_n\}$ is a solution of the recurrence relation

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k}$$

if and only if

$$\begin{aligned} a_n = {} & (\alpha_{1,0} + \alpha_{1,1} n + \cdots + \alpha_{1,m_1-1} n^{m_1-1}) r_1^n \\ & + (\alpha_{2,0} + \alpha_{2,1} n + \cdots + \alpha_{2,m_2-1} n^{m_2-1}) r_2^n \\ & + \cdots + (\alpha_{t,0} + \alpha_{t,1} n + \cdots + \alpha_{t,m_t-1} n^{m_t-1}) r_t^n \end{aligned}$$

for $n = 0, 1, 2, \ldots$, where $\alpha_{i,j}$ are constants for $1 \leq i \leq t$ and $0 \leq j \leq m_i - 1$.

The following example illustrates how Theorem 4 is used to find the general form of a solution of a linear homogeneous recurrence relation when the characteristic equation has several repeated roots.

EXAMPLE 7 Suppose that the roots of the characteristic equation of a linear homogeneous recurrence relation are 2, 2, 2, 5, 5, and 9 (that is, there are three roots, the root 2 with multiplicity three, the root 5 with multiplicity two, and the root 9 with multiplicity one). What is the form of the general solution?

Solution: By Theorem 4, the general form of the solution is

$$(\alpha_{1,0} + \alpha_{1,1} n + \alpha_{1,2} n^2) 2^n - (\alpha_{2,0} + \alpha_{2,1} n) 5^n + \alpha_{3,0} 9^n. \qquad \blacksquare$$

We now illustrate the use of Theorem 4 to solve a linear homogeneous recurrence relation with constant coefficients when the characteristic equation has a root of multiplicity three.

EXAMPLE 8 Find the solution to the recurrence relation

$$a_n = -3a_{n-1} - 3a_{n-2} - a_{n-3}$$

with initial conditions $a_0 = 1$, $a_1 = -2$, and $a_2 = -1$.

Solution: The characteristic equation of this recurrence relation is

$$r^3 + 3r^2 + 3r + 1 = 0.$$

Since $r^3 + 3r^2 + 3r + 1 = (r + 1)^3$, there is a single root $r = -1$ of multiplicity three of the characteristic equation. By Theorem 4 the solutions of this recurrence relation are of the form

$$a_n = \alpha_{1,0}(-1)^n + \alpha_{1,1} n(-1)^n + \alpha_{1,2} n^2 (-1)^n.$$

To find the constants $\alpha_{1,0}$, $\alpha_{1,1}$ and $\alpha_{1,2}$, use the initial conditions. This gives

$$\begin{aligned} a_0 &= 1 = \alpha_{1,0}, \\ a_1 &= -2 = -\alpha_{1,0} - \alpha_{1,1} - \alpha_{1,2}, \\ a_2 &= -1 = \alpha_{1,0} + 2\alpha_{1,1} + 4\alpha_{1,2}. \end{aligned}$$

The simultaneous solution of these three equations is $\alpha_{1,0} = 1$, $\alpha_{1,1} = 3$, and $\alpha_{1,2} = -2$. Hence, the unique solution to this recurrence relation and the given initial conditions is the sequence $\{a_n\}$ with

$$a_n = (1 + 3n - 2n^2)(-1)^n.$$

■

LINEAR NONHOMOGENEOUS RECURRENCE RELATIONS WITH CONSTANT COEFFICIENTS

We have seen how to solve linear homogeneous recurrence relations with constant coefficients. Is there a relatively simple technique for solving a linear, but not homogeneous, recurrence relation with constant coefficients, such as $a_n = 3a_{n-1} + 2n$? We will see that the answer is yes for certain families of such recurrence relations.

The recurrence relation $a_n = 3a_{n-1} + 2n$ is an example of a **linear nonhomogeneous recurrence relation with constant coefficients**, that is, a recurrence relation of the form

$$a_n = c_1a_{n-1} + c_2a_{n-2} + \cdots + c_ka_{n-k} + F(n),$$

where $c_1, c_2, \ldots, c_k$ are real numbers and $F(n)$ is a function not identically zero depending only on n. The recurrence relation

$$a_n = c_1a_{n-1} + c_2a_{n-2} + \cdots + c_ka_{n-k}$$

is called the **associated homogeneous recurrence relation**. It plays an important role in the solution of the nonhomogeneous recurrence relation.

EXAMPLE 9 Each of the recurrence relations $a_n = a_{n-1} + 2^n$, $a_n = a_{n-1} + a_{n-2} + n^2 + n + 1$, $a_n = 3a_{n-1} + n3^n$, and $a_n = a_{n-1} + a_{n-2} + a_{n-3} + n!$ is a linear nonhomogeneous recurrence relation with constant coefficients. The associated linear homogeneous recurrence relations are $a_n = a_{n-1}$, $a_n = a_{n-1} + a_{n-2}$, $a_n = 3a_{n-1}$, and $a_n = a_{n-1} + a_{n-2} + a_{n-3}$, respectively. ■

The key fact about linear nonhomogeneous recurrence relations with constant coefficients is that every solution is the sum of a particular solution and a solution of the associated linear homogeneous recurrence relation, as the following theorem shows.

THEOREM 5 **If $\{a_n^{(p)}\}$ is a particular solution of the nonhomogeneous linear recurrence relation with constant coefficients**

$$a_n = c_1a_{n-1} + c_2a_{n-2} + \cdots + c_ka_{n-k} + F(n),$$

then every solution is of the form $\{a_n^{(p)} + a_n^{(h)}\}$, where $\{a_n^{(h)}\}$ is a solution of the associated homogeneous recurrence relation

$$a_n = c_1a_{n-1} + c_2a_{n-2} + \cdots + c_ka_{n-k}.$$

Proof: Since $\{a_n^{(p)}\}$ is a particular solution of the nonhomogeneous recurrence relation, we know that

$$a_n^{(p)} = c_1 a_{n-1}^{(p)} + c_2 a_{n-2}^{(p)} + \cdots + c_k a_{n-k}^{(p)} + F(n).$$

Now suppose that $\{b_n\}$ is a second solution of the nonhomogeneous recurrence relation, so that

$$b_n = c_1 b_{n-1} + c_2 b_{n-2} + \cdots + c_k b_{n-k} + F(n).$$

Subtracting the first of these two equations from the second shows that

$$b_n - a_n^{(p)} = c_1(b_{n-1} - a_{n-1}^{(p)}) + c_2(b_{n-2} - a_{n-2}^{(p)}) + \cdots + c_k(b_{n-k} - a_{n-k}^{(p)}).$$

It follows that $\{b_n - a_n^{(p)}\}$ is a solution of the associated homogeneous linear recurrence, say, $\{a_n^{(h)}\}$. Consequently, $b_n = a_n^{(p)} + a_n^{(h)}$ for all n. □

By Theorem 5, we see that the key to solving nonhomogeneous recurrence relations with constant coefficients is finding a particular solution. Then every solution is a sum of this solution and a solution of the associated homogeneous recurrence relation. Although there is no general method for finding such a solution that works for every function $F(n)$, there are techniques that work for certain types of functions $F(n)$, such as polynomials and powers of constants. This is illustrated in Examples 10 and 11.

EXAMPLE 10 Find all solutions of the recurrence relation $a_n = 3a_{n-1} + 2n$. What is the solution with $a_1 = 3$?

Solution: To solve this linear homogeneous recurrence relation with constant coefficients, we need to solve its associated linear homogeneous equation and to find a particular solution for the given nonhomogeneous equation. The associated linear homogeneous equation is $a_n = 3a_{n-1}$. Its solutions are $a_n^{(h)} = \alpha 3^n$, where α is a constant.

We now find a particular solution. Since $F(n) = 2n$ is a polynomial in n of degree one, one reasonable trial solution is a linear function in n, say, $p_n = cn + d$, where c and d are constants. To determine whether there are any solutions of this form, suppose that $p_n = cn + d$ is such a solution. Then the equation $a_n = 3a_{n-1} + 2n$ becomes $cn + d = 3(c(n-1) + d) + 2n$. Simplifying and combining like terms gives $(2 + 2c)n + (2d - 3c) = 0$. It follows that $cn + d$ is a solution if and only if $2 + 2c = 0$ and $2d - 3c = 0$. This shows that $cn + d$ is a solution if and only if $c = -1$ and $d = -3/2$. Consequently, $a_n^{(p)} = -n - 3/2$ is a particular solution.

By Theorem 5 all solutions are of the form

$$a_n = a_n^{(p)} + a_n^{(h)} = -n - \frac{3}{2} + \alpha \cdot 3^n,$$

where α is a constant.

To find the solution with $a_1 = 3$, let $n = 1$ in the formula we obtained for the general solution. We find that $3 = -1 - 3/2 + 3\alpha$, which implies that $\alpha = 11/6$. The solution we seek is $a_n = -n - 3/2 + (11/6)3^n$. ■

EXAMPLE 11 Find all solutions of the recurrence relation

$$a_n = 5a_{n-1} - 6a_{n-2} + 7^n.$$

Solution: This is a linear nonhomogeneous recurrence relation. The solutions of its associated homogeneous recurrence relation

$$a_n = 5a_{n-1} - 6a_{n-2}$$

are $a_n^{(h)} = \alpha_1 \cdot 3^n + \alpha_2 \cdot 2^n$, where α_1 and α_2 are constants. Since $F(n) = 7^n$, a reasonable trial solution is $a_n^{(p)} = C \cdot 7^n$, where C is a constant. Substituting the terms of this sequence into the recurrence relation implies that $C \cdot 7^n = 5C \cdot 7^{n-1} - 6C \cdot 7^{n-2} + 7^n$. Factoring out 7^{n-2}, this equation becomes $49C = 35C - 6C + 49$, which implies that $20C = 49$, or that $C = 49/20$. Hence, $a_n^{(p)} = (49/20)7^n$ is a particular solution. By Theorem 5, all solutions are of the form

$$a_n = \alpha_1 \cdot 3^n + \alpha_2 \cdot 2^n + (49/20)7^n.$$ ■

In Examples 10 and 11, we made an educated guess that there are solutions of a particular form. In both cases we were able to find particular solutions. This was not an accident. Whenever $F(n)$ is the product of a polynomial in n and the nth power of a constant, we know exactly what form a particular solution has, as stated in Theorem 6. We leave the proof of Theorem 6 as a challenging exercise for the reader.

THEOREM 6 **Suppose that $\{a_n\}$ satisfies the linear nonhomogeneous recurrence relation**

$$a_n = c_1a_{n-1} + c_2a_{n-2} + \cdots + c_ka_{n-k} + F(n),$$

where $c_1, c_2, \ldots, c_k$ are real numbers and

$$F(n) = (b_tn^t + b_{t-1}n^{t-1} + \cdots + b_1n + b_0)s^n,$$

where $b_0, b_1, \ldots, b_t$ and s are real numbers. When s is not a root of the characteristic equation of the associated linear homogeneous recurrence relation, there is a particular solution of the form

$$(p_tn^t + p_{t-1}n^{t-1} + \cdots + p_1n + p_0)s^n.$$

When s is a root of this characteristic equation and its multiplicity is m, there is a particular solution of the form

$$n^m(p_tn^t + p_{t-1}n^{t-1} + \cdots + p_1n + p_0)s^n.$$

Note that the factor n^m in the case when s is a root of multiplicity m of the characteristic equation of the associated linear homogeneous recurrence relation ensures that the proposed particular solution will not already be a solution of the associated linear homogeneous recurrence relation. We next provide an example to illustrate the form of a particular solution provided by Theorem 6.

EXAMPLE 12 What form does a particular solution of the linear nonhomogeneous recurrence relation $a_n = 6a_{n-1} - 9a_{n-2} + F(n)$ have when $F(n) = 3^n$, $F(n) = n3^n$, $F(n) = n^22^n$, and $F(n) = (n^2 + 1)3^n$?

Solution: The associated linear homogeneous recurrence relation is $a_n = 6a_{n-1} - 9a_{n-2}$. Its characteristic equation, $r^2 - 6r + 9 = (r - 3)^2 = 0$, has a single root, 3, of multiplicity two. To apply Theorem 6, with $F(n)$ of the form $P(n)s^n$, where $P(n)$ is a polynomial and s is a constant, we need to ask whether s is a root of this characteristic equation.

Since $s = 3$ is a root with multiplicity $m = 2$ but $s = 2$ is not a root, Theorem 6 tells us that a particular solution has the form $p_0n^23^n$ if $F(n) = 3^n$, the form $n^2(p_1n + p_0)3^n$ if $F(n) = n3^n$, the form $(p_2n^2 + p_1n + p_0)2^n$ if $F(n) = n^22^n$, and the form $n^2(p_2n^2 + p_1n + p_0)3^n$ if $F(n) = (n^2 + 1)3^n$. ■

Care must be taken when $s = 1$ when solving recurrence relations of the type covered by Theorem 6. In particular, to apply this theorem with $F(n) = b_tn^t + b_{t-1}n^{t-1} + \cdots + b_1n + b_0$, the parameter s takes the value $s = 1$ (even though the term 1^n does not explicitly appear). By the theorem, the form of the solution then depends on whether 1 is a root of the characteristic equation of the associated linear homogeneous recurrence relation. This is illustrated in Example 13, which shows how Theorem 6 can be used to find a formula for the sum of the first n positive integers.

EXAMPLE 13 Let a_n be the sum of the first n positive integers, so that

$$a_n = \sum_{k=1}^{n} k.$$

Note that a_n satisfies the linear nonhomogeneous recurrence relation

$$a_n = a_{n-1} + n.$$

(To obtain a_n, the sum of the first n positive integers, from a_{n-1}, the sum of the first $n - 1$ positive integers, we add n.) Note that the initial condition is $a_1 = 1$.

The associated linear homogeneous recurrence relation for a_n is

$$a_n = a_{n-1}.$$

The solutions of this homogeneous recurrence relation are given by $a_n^{(h)} = c(1)^n = c$, where c is a constant. To find all solutions of $a_n = a_{n-1} + n$, we need find only a single particular solution. By Theorem 6, since $F(n) = n = n \cdot (1)^n$ and $s = 1$ is a root of degree one of the characteristic equation of the associated linear homogeneous recurrence relations, there is a particular solution of the form $n(p_1n + p_0) = p_1n^2 + p_0n$.

Inserting this into the recurrence relation gives $p_1n^2 + p_0n = p_1(n-1)^2 + p_0(n-1) + n$. Simplifying, we see that $n(2p_1 - 1) + (p_0 - p_1) = 0$, which means that $2p_1 - 1 = 0$ and $p_0 - p_1 = 0$, so $p_0 = p_1 = 1/2$. Hence,

$$a_n^{(p)} = \frac{n^2}{2} + \frac{n}{2} = \frac{n(n+1)}{2}$$

is a particular solution. Hence, all solutions of the original recurrence relation $a_n = a_{n-1} + n$ are given by $a_n = a_n^{(h)} + a_n^{(p)} = c + n(n+1)/2$. Since $a_1 = 1$, we have $1 = a_1 = c + 1 \cdot 2/2 = c + 1$, so $c = 0$. It follows that $a_n = n(n+1)/2$. (This is the same formula found in Exercise 22 in Section 1.7.) ■

Exercises

1. Determine which of the following are linear homogeneous recurrence relations with constant coefficients. Also, find the degree of those that are.

a) $a_n = 3a_{n-1} + 4a_{n-2} + 5a_{n-3}$

b) $a_n = 2na_{n-1} + a_{n-2}$

c) $a_n = a_{n-1} + a_{n-4}$

d) $a_n = a_{n-1} + 2$ e) $a_n = a_{n-1}^2 + a_{n-2}$
f) $a_n = a_{n-2}$
g) $a_n = a_{n-1} + n$

2. Determine which of the following are linear homogeneous recurrence relations with constant coefficients. Also, find the degree of those that are.
a) $a_n = 3a_{n-2}$ b) $a_n = 3$
c) $a_n = a_{n-1}^2$ d) $a_n = a_{n-1} + 2a_{n-3}$
e) $a_n = a_{n-1}/n$ f) $a_n = a_{n-1} + a_{n-2} + n + 3$
g) $a_n = 4a_{n-2} + 5a_{n-4} + 9a_{n-7}$

3. Solve the following recurrence relations together with the initial conditions given.
a) $a_n = 2a_{n-1}$ for $n \geq 1$, $a_0 = 3$
b) $a_n = a_{n-1}$ for $n \geq 1$, $a_0 = 2$
c) $a_n = 5a_{n-1} - 6a_{n-2}$ for $n \geq 2$, $a_0 = 1$, $a_1 = 0$
d) $a_n = 4a_{n-1} - 4a_{n-2}$ for $n \geq 2$, $a_0 = 6$, $a_1 = 8$
e) $a_n = -4a_{n-1} - 4a_{n-2}$ for $n \geq 2$, $a_0 = 0$, $a_1 = 1$
f) $a_n = 4a_{n-2}$ for $n \geq 2$, $a_0 = 0$, $a_1 = 4$
g) $a_n = a_{n-2}/4$ for $n \geq 2$, $a_0 = 1$, $a_1 = 0$

4. Solve the following recurrence relations together with the initial conditions given.
a) $a_n = a_{n-1} + 6a_{n-2}$ for $n \geq 2$, $a_0 = 3$, $a_1 = 6$
b) $a_n = 7a_{n-1} - 10a_{n-2}$ for $n \geq 2$, $a_0 = 2$, $a_1 = 1$
c) $a_n = 6a_{n-1} - 8a_{n-2}$ for $n \geq 2$, $a_0 = 4$, $a_1 = 10$
d) $a_n = 2a_{n-1} - a_{n-2}$ for $n \geq 2$, $a_0 = 4$, $a_1 = 1$
e) $a_n = a_{n-2}$ for $n \geq 2$, $a_0 = 5$, $a_1 = -1$
f) $a_n = -6a_{n-1} - 9a_{n-2}$ for $n \geq 2$, $a_0 = 3$, $a_1 = -3$
g) $a_{n-2} = -4a_{n+1} + 5a_n$ for $n \geq 0$, $a_0 = 2$, $a_1 = 8$

5. How many different messages can be transmitted in n microseconds using the two signals described in Exercise 29 of Section 5.1?

6. How many different messages can be transmitted in n microseconds using three different signals if one signal requires 1 microsecond for transmittal, the other two signals require 2 microseconds each for transmittal, and a signal in a message is followed immediately by the next signal?

7. In how many ways can a $2 \times n$ rectangular board be tiled using 1×2 and 2×2 pieces?

8. A model for the number of lobsters caught per year is based on the assumption that the number of lobsters caught in a year is the average of the number caught in the two previous years.
a) Find a recurrence relation for $\{L_n\}$, where L_n is the number of lobsters caught in year n, under the assumption for this model.
b) Find L_n if 100,000 lobsters were caught in year 1 and 300,000 were caught in year 2.

9. A deposit of \$100,000 is made to an investment fund at the beginning of a year. On the last day of each year two dividends are awarded. The first dividend is 20% of the amount in the account during that year. The second dividend is 45% of the amount in the account in the previous year.
a) Find a recurrence relation for $\{P_n\}$, where P_n is the amount in the account at the end of n years if no money is ever withdrawn.
b) How much is in the account after n years if no money has been withdrawn?

***10.** Prove Theorem 2.

11. The **Lucas numbers** satisfy the recurrence relation

$$L_n = L_{n-1} + L_{n-2},$$

and the initial conditions $L_0 = 2$ and $L_1 = 1$.
a) Show that $L_n = f_{n-1} + f_{n+1}$ for $n = 2, 3, \ldots$, where f_n is the nth Fibonacci number.
b) Find an explicit formula for the Lucas numbers.

12. Find the solution to $a_n = 2a_{n-1} + a_{n-2} - 2a_{n-3}$ for $n = 3, 4, 5, \ldots$, with $a_0 = 3$, $a_1 = 6$, and $a_2 = 0$.

13. Find the solution to $a_n = 7a_{n-2} + 6a_{n-3}$ with $a_0 = 9$, $a_1 = 10$, and $a_2 = 32$.

14. Find the solution to $a_n = 5a_{n-2} - 4a_{n-4}$ with $a_0 = 3$, $a_1 = 2$, $a_2 = 6$, and $a_3 = 8$.

15. Find the solution to $a_n = 2a_{n-1} + 5a_{n-2} - 6a_{n-3}$ with $a_0 = 7$, $a_1 = -4$, and $a_2 = 8$.

***16.** Prove Theorem 3.

17. Prove the following identity relating the Fibonacci numbers and the binomial coefficients:

$$f_{n+1} = C(n, 0) + C(n-1, 1) + \cdots + C(n-k, k)$$

where n is a positive integer and $k = \lfloor n/2 \rfloor$. [*Hint:* Let $a_n = C(n, 0) + C(n-1, 1) + \cdots + C(n-k, k)$. Show that the sequence $\{a_n\}$ satisfies the same recurrence relation and initial conditions satisfied by the sequence of Fibonacci numbers.]

18. Solve the recurrence relation $a_n = 6a_{n-1} - 12a_{n-2} + 8a_{n-3}$ with $a_0 = -5$, $a_1 = 4$, and $a_2 = 88$.

19. Solve the recurrence relation $a_n = -3a_{n-1} - 3a_{n-2} - a_{n-3}$ with $a_0 = 5$, $a_1 = -9$, and $a_2 = 15$.

20. Find the general form of the solutions of the recurrence relation $a_n = 8a_{n-2} - 16a_{n-4}$.

21. What is the general form of the solutions of a linear homogeneous recurrence relation if its characteristic equation has roots 1, 1, 1, 1, −2, −2, −2, 3, 3, −4?

22. What is the general form of the solutions of a linear homogeneous recurrence relation if its characteristic equation has the roots −1, −1, −1, 2, 2, 5, 5, 7?

23. Consider the nonhomogeneous linear recurrence relation $a_n = 3a_{n-1} + 2^n$.
a) Show that $a_n = -2^{n+1}$ is a solution of this recurrence relation.
b) Use Theorem 5 to find all solutions of this recurrence relation.
c) Find the solution with $a_0 = 1$.

24. Consider the nonhomogeneous linear recurrence relation $a_n = 2a_{n-1} + 2^n$.
a) Show that $a_n = n2^n$ is a solution of this recurrence relation.

b) Use Theorem 5 to find all solutions of this recurrence relation.

c) Find the solution with $a_0 = 2$.

25. a) Determine values of the constants A and B so that $a_n = An + B$ is a solution of the recurrence relation $a_n = 2a_{n-1} + n + 5$.

b) Use Theorem 5 to find all the solutions of this recurrence relation.

c) Find the solution of this recurrence relation with $a_0 = 4$.

26. What is the general form of the particular solution of the linear nonhomogeneous recurrence relation $a_n = 6a_{n-1} - 12a_{n-2} + 8a_{n-3} + F(n)$ if

a) $F(n) = n^2$?

b) $F(n) = 2^n$?

c) $F(n) = n2^n$?

d) $F(n) = (-2)^n$?

e) $F(n) = n^2 2^n$?

f) $F(n) = n^3(-2)^n$?

g) $F(n) = 3$?

27. What is the general form of the particular solution of the linear nonhomogeneous recurrence relation $a_n = 8a_{n-2} - 16a_{n-4} + F(n)$ if

a) $F(n) = n^3$?

b) $F(n) = (-2)^n$?

c) $F(n) = n2^n$?

d) $F(n) = n^2 4^n$?

e) $F(n) = (n^2 - 2)(-2)^n$?

f) $F(n) = n^4 2^n$?

g) $F(n) = 2$?

28. a) Find all solutions of the recurrence relation $a_n = 2a_{n-1} + 2n^2$.

b) Find the solution of the recurrence relation in part (a) with initial condition $a_1 = 4$.

29. a) Find all solutions of the recurrence relation $a_n = 2a_{n-1} + 3^n$.

b) Find the solution of the recurrence relation in part (a) with initial condition $a_1 = 5$.

30. a) Find all solutions of the recurrence relation $a_n = -5a_{n-1} - 6a_{n-2} + 42 \cdot 4^n$.

b) Find the solution of this recurrence relation with $a_1 = 56$ and $a_2 = 278$.

31. Find all solutions of the recurrence relation $a_n = 5a_{n-1} - 6a_{n-2} + 2^n + 3n$. (*Hint:* Look for a particular solution of the form $qn2^n + p_1 n + p_2$, where q, p_1, and p_2 are constants.)

32. Find the solution of the recurrence relation $a_n = 2a_{n-1} + 3 \cdot 2^n$.

33. Find all solutions of the recurrence relation $a_n = 4a_{n-1} - 4a_{n-2} + (n+1)2^n$.

34. Find all solutions of the recurrence relation $a_n = 7a_{n-1} - 16a_{n-2} + 12a_{n-3} + n4^n$ with $a_0 = -2$, $a_1 = 0$, and $a_2 = 5$.

35. Find the solution of the recurrence relation $a_n = 4a_{n-1} - 3a_{n-2} + 2^n + n + 3$ with $a_0 = 1$ and $a_1 = 4$.

36. Let a_n be the sum of the first n perfect squares, that is, $a_n = \sum_{k=1}^{n} k^2$. Show that the sequence $\{a_n\}$ satisfies the linear nonhomogeneous recurrence relation $a_n = a_{n-1} + n^2$ and the initial condition $a_1 = 1$. Use Theorem 6 to determine a formula for a_n by solving this recurrence relation.

37. Let a_n be the sum of the first n triangular numbers, that is, $a_n = \sum_{k=1}^{n} t_k$, where $t_k = k(k+1)/2$. Show that $\{a_n\}$ satisfies the linear nonhomogeneous recurrence relation $a_n = a_{n-1} + n(n+1)/2$ and the initial condition $a_1 = 1$. Use Theorem 6 to determine a formula for a_n by solving this recurrence relation.

38. a) Find the characteristic roots of the linear homogeneous recurrence relation $a_n = 2a_{n-1} - 2a_{n-2}$. (*Note:* These are complex numbers.)

b) Find the solution of the recurrence relation in part (a) with $a_0 = 1$ and $a_1 = 2$.

***39. a)** Find the characteristic roots of the linear homogeneous recurrence relation $a_n = a_{n-4}$. (*Note:* These include complex numbers.)

b) Find the solution of the recurrence relation in part (a) with $a_0 = 1$, $a_1 = 0$, $a_2 = -1$, and $a_3 = 1$.

***40.** Solve the simultaneous recurrence relations

$$a_n = 3a_{n-1} + 2b_{n-1}$$
$$b_n = a_{n-1} + 2b_{n-1}$$

with $a_0 = 1$ and $b_0 = 2$.

***41. a)** Use the formula found in Example 4 for f_n, the nth Fibonacci number, to show that f_n is the integer closest to

$$\frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n.$$

b) Determine for which n f_n is greater than

$$\frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n$$

and for which n f_n is less than

$$\frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n.$$

42. Show that if $a_n = a_{n-1} + a_{n-2}$, $a_0 = s$ and $a_1 = t$, where s and t are constants, then $a_n = sf_{n-1} + tf_n$ for all positive integers n.

43. Express the solution of the linear nonhomogenous recurrence relation $a_n = a_{n-1} + a_{n-2} + 1$ for $n \geq 2$ where $a_0 = 0$ and $a_1 = 1$ in terms of the Fibonacci numbers. (*Hint:* Let $b_n = a_n + 1$ and apply Exercise 42 to the sequence b_n.)

***44.** (Linear algebra required) Let $\mathbf{A}_n$ be the $n \times n$ matrix with 2s on its main diagonal, 1s in all positions next to a diagonal element, and 0s everywhere else. Find a recurrence relation for d_n, the determinant of $\mathbf{A}_n$. Solve this recurrence relation to find a formula for d_n.

45. Suppose that each pair of a genetically engineered species of rabbits left on an island produces two new pairs of rabbits at the age of 1 month and six new pairs of rabbits at the age of 2 months and every month afterward. None of the rabbits ever die or leave the island.

a) Find a recurrence relation for the number of pairs of rabbits on the island n months after one newborn pair is left on the island.

b) By solving the recurrence relation in (a) determine the number of pairs of rabbits on the island n months after one pair is left on the island.

46. Suppose that there are two goats on an island initially. The number of goats on the island doubles every year by natural reproduction, and some goats are either added or removed each year.

a) Construct a recurrence relation for the number of goats on the island at the start of the nth year, assuming that during each year an extra 100 goats are put on the island.

b) Solve the recurrence relation from part (a) to find the number of goats on the island at the start of the nth year.

c) Construct a recurrence relation for the number of goats on the island at the start of the nth year, assuming that n goats are removed during the nth year for each $n \geq 3$.

d) Solve the recurrence relation in part (c) for the number of goats on the island at the start of the nth year.

47. A new employee at an exciting new software company starts with a salary of \$50,000 and is promised that at the end of each year her salary will be double her salary the previous year, with an extra increment of \$10,000 for each year she has been with the company.

a) Construct a recurrence relation for her salary for her nth year of employment.

b) Solve this recurrence relation to find her salary for her nth year of employment.

Some linear recurrence relations do not have constant coefficients which can be systematically solved. This is the case for recurrence relations of the form $f(n)a_n = g(n)a_{n-1} + h(n)$. Exercises 48–50 illustrate this.

***48.** a) Show that the recurrence relation

$$f(n)a_n = g(n)a_{n-1} + h(n),$$

for $n \geq 1$, and with $a_0 = C$, can be reduced to a recurrence relation of the form

$$b_n = b_{n-1} + Q(n)h(n),$$

where $b_n = g(n+1)Q(n+1)a_n$, with

$$Q(n) = (f(1)f(2)\cdots f(n-1))/(g(1)g(2)\cdots g(n)).$$

b) Use part (a) to solve the original recurrence relation to obtain

$$a_n = \frac{C + \sum_{i=1}^{n} Q(i)h(i)}{g(n+1)Q(n+1)}.$$

***49.** Use Exercise 48 to solve the recurrence relation $(n+1)a_n = (n+3)a_{n-1} + n$, for $n \geq 1$, with $a_0 = 1$.

50. It can be shown that the average number of comparisons made by the quick sort algorithm (described in the exercise set of Section 8.4), when sorting n elements in random order, satisfies the recurrence relation

$$C_n = n + 1 + \frac{2}{n}\sum_{k=0}^{n-1} C_k$$

for $n = 1, 2, \ldots$, with initial condition $C_0 = 0$.

a) Show that $\{C_n\}$ also satisfies the recurrence relation $nC_n = (n+1)C_{n-1} + 2n$ for $n = 1, 2, \ldots$.

b) Use Exercise 48 to solve the recurrence relation in part (a) to find an explicit formula for C_n.

****51.** Prove Theorem 4.

****52.** Prove Theorem 6.

5.3

Divide-and-Conquer Relations

INTRODUCTION

Many recursive algorithms take a problem with given input and divide it into one or more smaller problems. This reduction is successively applied until the solutions of the smaller problems can be found quickly. For instance, we perform a binary search by reducing the search for an element in a list to the search for this element in a list half as long. We successively apply this reduction until one element is left. Another example of this type of recursive algorithm is a procedure for multiplying integers that reduces the problem of the multiplication of two integers to three multiplications of pairs of integers with half as many bits. This reduction is successively applied until integers with one bit are obtained. These procedures are called **divide-and-conquer** algorithms. In this section the recurrence relations that arise in the analysis of the complexity of these algorithms will be studied.

DIVIDE-AND-CONQUER RELATIONS

Suppose that an algorithm splits a problem of size n into a subproblems, where each subproblem is of size n/b (for simplicity, suppose that b divides n; in reality, the smaller problems are often of size equal to the nearest integer either less than or equal to, or greater than or equal to, n/b). Also, suppose that a total of $g(n)$ extra operations are required when this split of a problem of size n into smaller problems is made. Then, if $f(n)$ represents the number of operations required to solve the problem, it follows that f satisfies the recurrence relation

$$f(n) = af(n/b) + g(n).$$

This is called a **divide-and-conquer** recurrence relation.

EXAMPLE 1 We introduced a binary search algorithm in Section 2.1. This binary search algorithm reduces the search for an element in a search sequence of size n to the binary search for this element in a search sequence of size $n/2$, when n is even. (Hence, the problem of size n has been reduced to *one* problem of size $n/2$.) Two comparisons are needed to implement this reduction (one to determine which half of the list to use and the other to determine whether any terms of the list remain). Hence, if $f(n)$ is the number of comparisons required to search for an element in a search sequence of size n, then $f(n) = f(n/2) + 2$ when n is even. ■

EXAMPLE 2 Consider the following algorithm for locating the minimum and maximum elements of a sequence $a_1, a_2, \ldots, a_n$. If $n = 1$, then a_1 is the maximum and the minimum. If $n > 1$, split the sequence into two sequences, either where both have the same number of elements or where one of the sets has one element more than the other. The problem is reduced to finding the maximum and minimum of each of the two smaller sequences. The solution to the original problem results from the comparison of the separate maxima and minima of the two smaller sets to obtain the overall maximum and minimum.

Let $f(n)$ be the total number of comparisons needed to find the minimum and maximum elements of the set with n elements. We have shown that a problem of size n can be reduced into two problems of size $n/2$, when n is even, using two comparisons, one to compare the minima of the two sets and the other to compare the maxima of the two sets. This gives the recurrence relation $f(n) = 2f(n/2) + 2$ when n is even. ■

EXAMPLE 3 Surprisingly, there are more efficient algorithms than the conventional algorithm (described in Section 2.4) for multiplying integers. One of these algorithms, which uses a divide-and-conquer technique, will be described here. This fast multiplication algorithm proceeds by splitting each of two $2n$-bit integers into two blocks each with n bits. Then, the original multiplication is reduced from the multiplication of two $2n$-bit integers to three multiplications of n-bit integers, plus shifts and additions.

Suppose that a and b are integers with binary expansions of length $2n$ (add initial bits of zero in these expansions if necessary to make them the same length). Let

$$a = (a_{2n-1}a_{2n-2}\cdots a_1a_0)_2 \qquad \text{and} \qquad b = (b_{2n-1}b_{2n-2}\cdots b_1b_0)_2.$$

Let

$$a = 2^nA_1 + A_0, \qquad b = 2^nB_1 + B_0,$$

where

$$\begin{aligned} A_1 &= (a_{2n-1}\cdots a_{n+1}a_n)_2, & A_0 &= (a_{n-1}\cdots a_1a_0)_2, \\ B_1 &= (b_{2n-1}\cdots b_{n+1}b_n)_2, & B_0 &= (b_{n-1}\cdots b_1b_0)_2. \end{aligned}$$

The algorithm for fast multiplication of integers is based on the identity

$$ab = (2^{2n} + 2^n)A_1B_1 + 2^n(A_1 - A_0)(B_0 - B_1) + (2^n + 1)A_0B_0.$$

The important fact about this identity is that it shows that the multiplication of two $2n$-bit integers can be carried out using three multiplications of n-bit integers, together with additions, subtractions, and shifts. This shows that if $f(n)$ is the total number of bit operations needed to multiply two n-bit integers, then

$$f(2n) = 3f(n) + Cn.$$

The reasoning behind this equation is as follows. The three multiplications of n-bit integers are carried out using $3f(n)$-bit operations. Each of the additions, subtractions, and shifts uses a constant multiple of n-bit operations, and Cn represents the total number of bit operations used by these operations. ■

EXAMPLE 4 There are algorithms that multiply two $n \times n$ matrices, when n is even, using seven multiplications each of two $(n/2) \times (n/2)$ matrices and 15 additions of $(n/2) \times (n/2)$ matrices. Hence, if $f(n)$ is the number of operations (multiplications and additions) used, it follows that

$$f(n) = 7f(n/2) + 15n^2/4$$

when n is even. ■

As Examples 1–4 show, recurrence relations of the form $f(n) = af(n/b) + g(n)$ arise in many different situations. It is possible to derive estimates of the size of functions that satisfy such recurrence relations. Suppose that f satisfies this recurrence relation whenever n is divisible by b. Let $n = b^k$, where k is a positive integer. Then

$$\begin{aligned} f(n) &= af(n/b) + g(n) \\ &= a^2f(n/b^2) + ag(n/b) + g(n) \\ &= a^3f(n/b^3) + a^2g(n/b^2) + ag(n/b) + g(n) \\ &\ \vdots \\ &= a^kf(n/b^k) + \sum_{j=0}^{k-1} a^jg(n/b^j). \end{aligned}$$

Since $n/b^k = 1$, it follows that

$$f(n) = a^k f(1) + \sum_{j=0}^{k-1} a^j g(n/b^j).$$

We can use this equation for $f(n)$ to estimate the size of functions that satisfy divide-and-conquer relations.

THEOREM 1 **Let f be an increasing function that satisfies the recurrence relation**

$$f(n) = af(n/b) + c$$

whenever n is divisible by b, where $a \geq 1$, b is an integer greater than 1, and c is a positive real number. Then

$$f(n) = \begin{cases} O(n^{\log_b a}) & \text{if } a > 1, \\ O(\log n) & \text{if } a = 1. \end{cases}$$

Proof: First let $n = b^k$. From the expression for $f(n)$ obtained in the discussion preceding the theorem, with $g(n) = c$, we have

$$f(n) = a^k f(1) + \sum_{j=0}^{k-1} a^j c = a^k f(1) + c\sum_{j=0}^{k-1} a^j.$$

First consider the case when $a = 1$. Then

$$f(n) = f(1) + ck.$$

Since $n = b^k$, we have $k = \log_b n$. Hence

$$f(n) = f(1) + c\log_b n.$$

When n is not a power of b, we have $b^k < n < b^{k+1}$, for a positive integer k. Since f is increasing, it follows that $f(n) \leq f(b^{k+1}) = f(1) + c(k+1) = (f(1) + c) + ck \leq (f(1) + c) + c\log_b n$. Therefore, in both cases, $f(n)$ is $O(\log n)$ when $a = 1$.

Now suppose that $a > 1$. First assume that $n = b^k$ where k is a positive integer. From the formula for the sum of terms of a geometric progression (Example 6 of Section 3.2), it follows that

$$\begin{aligned} f(n) &= a^k f(1) + c(a^k - 1)/(a - 1) \\ &= a^k[f(1) + c/(a - 1)] - c/(a - 1) \\ &= C_1 n^{\log_b a} + C_2, \end{aligned}$$

since $a^k = a^{\log_b n} = n^{\log_b a}$ (see Exercise 4 in Appendix 1), where $C_1 = [f(1)+c/(a-1)]$ and $C_2 = -c/(a-1)$.

Now suppose that n is not a power of b. Then $b^k < n < b^{k+1}$ where k is a nonnegative integer. Since f is increasing,

$$\begin{aligned} f(n) &\leq f(b^{k+1}) = C_1 a^{k+1} + C_2 \\ &\leq (C_1 a)a^{\log_b n} + C_2 \\ &\leq (C_1 a)n^{\log_b a} + C_2, \end{aligned}$$

since $k \leq \log_b n < k + 1$.

Hence, we have $f(n)$ is $O(n^{\log_b a})$. □

Remark: This proof gives an explicit formula for $f(n)$ where $n = b^k$.

The following examples illustrate how Theorem 1 is used.

EXAMPLE 5 Let $f(n) = 5f(n/2) + 3$ and $f(1) = 7$. Find $f(2^k)$ where k is a positive integer. Also, estimate $f(n)$ if f is an increasing function.

Solution: From the proof of Theorem 1, with $a = 5$, $b = 2$, and $c = 3$, we see that if $n = 2^k$, then

$$\begin{aligned} f(n) &= a^k[f(1) + c/(a-1)] + [-c/(a-1)] \\ &= 5^k[7 + (3/4)] - 3/4 \\ &= 5^k(31/4) - 3/4. \end{aligned}$$

Also, if $f(n)$ is increasing, Theorem 1 shows that $f(n)$ is $O(n^{\log_b a}) = O(n^{\log 5})$. ■

We can use Theorem 1 to estimate the computational complexity of the binary search algorithm and the algorithm given in Example 2 for locating the minimum and maximum of a sequence.

EXAMPLE 6 Estimate the number of comparisons used by a binary search.

Solution: In Example 1 it was shown that $f(n) = f(n/2) + 2$ when n is even, where f is the number of comparisons required to perform a binary search on a sequence of size n. Hence, from Theorem 1, it follows that $f(n)$ is $O(\log n)$. ■

EXAMPLE 7 Estimate the number of comparisons used to locate the maximum and minimum elements in a sequence using the algorithm given in Example 2.

Solution: In Example 2 we showed that $f(n) = 2f(n/2) + 2$, when n is even, where f is the number of comparisons needed by this algorithm. Hence, from Theorem 1, it follows that $f(n)$ is $O(n^{\log 2}) = O(n)$. ■

We will now state a more general, and more complicated, theorem that is useful in analyzing the complexity of divide-and-conquer algorithms.

THEOREM 2 **Let f be an increasing function that satisfies the recurrence relation**

$$f(n) = af(n/b) + cn^d$$

whenever $n = b^k$, where k is a positive integer, $a \geq 1$, b is an integer greater than 1, and c and d are positive real numbers. Then

$$f(n) = \begin{cases} O(n^d) & \text{if } a < b^d, \\ O(n^d \log n) & \text{if } a = b^d, \\ O(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

The proof of Theorem 2 is left for the reader as Exercises 17–21 at the end of this section.

EXAMPLE 8 Estimate the number of bit operations needed to multiply two n-bit integers using the fast multiplication algorithm.

Solution: Example 3 shows that $f(n) = 3f(n/2) + Cn$, when n is even, where $f(n)$ is the number of bit operations required to multiply two n-bit integers using the fast multiplication algorithm. Hence, from Theorem 2, it follows that $f(n)$ is $O(n^{\log 3})$. Note that $\log 3 \sim 1.6$. Since the conventional algorithm for multiplication uses $O(n^2)$ bit operations, the fast multiplication algorithm is a substantial improvement over the conventional algorithm in terms of time complexity for sufficiently large integers. ■

EXAMPLE 9 Estimate the number of multiplications and additions required to multiply two $n \times n$ matrices using the matrix multiplication algorithm referred to in Example 4.

Solution: Let $f(n)$ denote the number of additions and multiplications used by the algorithm mentioned in Example 4 to multiply two $n \times n$ matrices. We have $f(n) = 7f(n/2) + 15n^2/4$, when n is even. Hence, from Theorem 2, it follows that $f(n)$ is $O(n^{\log 7})$. Note that $\log 7 \sim 2.8$. Since the conventional algorithm for multiplying two $n \times n$ matrices uses $O(n^3)$ additions and multiplications, it follows that for sufficiently large integers n, this algorithm is substantially more efficient in time complexity than the conventional algorithm. ■

Exercises

1. How many comparisons are needed for a binary search in a set of 64 elements?

2. How many comparisons are needed to locate the maximum and minimum elements in a sequence with 128 elements using the algorithm in Example 2?

3. Multiply $(1110)_2$ and $(1010)_2$ using the fast multiplication algorithm.

4. Express the fast multiplication algorithm in pseudocode.

5. Determine a value for the constant C in Example 3 and use it to estimate the number of bit operations needed to multiply two 64-bit integers using the fast multiplication algorithm.

6. How many operations are needed to multiply two 32×32 matrices using the algorithm referred to in Example 4?

7. Suppose that $f(n) = f(n/3) + 1$ when n is divisible by 3, and $f(1) = 1$. Find

a) $f(3)$.
b) $f(27)$.
c) $f(729)$.

8. Suppose that $f(n) = 2f(n/2) + 3$ when n is even, and $f(1) = 5$. Find

a) $f(2)$.
b) $f(8)$.
c) $f(64)$.
d) $f(1024)$.

9. Suppose that $f(n) = f(n/5) + 3n^2$ when n is divisible by 5, and $f(1) = 4$. Find

a) $f(5)$.
b) $f(125)$.
c) $f(3125)$.

10. Find $f(n)$ when $n = 2^k$, where f satisfies the recurrence relation $f(n) = f(n/2) + 1$ with $f(1) = 1$.

11. Estimate the size of f in Exercise 10 if f is an increasing function.

12. Find $f(n)$ when $n = 3^k$, where f satisfies the recurrence relation $f(n) = 2f(n/3) + 4$ with $f(1) = 1$.

13. Estimate the size of f in Exercise 12 if f is an increasing function.

14. Suppose that there are $n = 2^k$ teams in an elimination tournament, where there are $n/2$ games in the first round, with the $n/2 = 2^{k-1}$ winners playing in the second round, and so on. Develop a recurrence relation for the number of rounds in the tournament.

15. How many rounds are in the elimination tournament described in Exercise 14 when there are 32 teams?

16. Solve the recurrence relation for the number of rounds in the tournament described in Exercise 14.

In Exercises 17–21, assume that f is an increasing function satisfying the recurrence relation $f(n) = af(n/b) + cn^d$, where $a \geq 1$, b is an integer greater than 1, and c and d are positive real numbers. These exercises supply a proof of Theorem 2.

***17.** Show that if $a = b^d$ and n is a power of b, then $f(n) = f(1)n^d + cn^d \log_b n$.

18. Use Exercise 17 to show that if $a = b^d$, then $f(n)$ is $O(n^d \log n)$.

***19.** Show that if $a \neq b^d$ and n is a power of b, then $f(n) = C_1 n^d + C_2 n^{\log_b a}$, where $C_1 = b^d c/(b^d - a)$ and $C_2 = f(1) + b^d c/(a - b^d)$.

20. Use Exercise 19 to show that if $a < b^d$, then $f(n)$ is $O(n^d)$.

21. Use Exercise 19 to show that if $a > b^d$, then $f(n)$ is $O(n^{\log_b a})$.

22. Find $f(n)$ when $n = 4^k$, where f satisfies the recurrence relation $f(n) = 5f(n/4) + 6n$, with $f(1) = 1$.

23. Estimate the size of f in Exercise 22 if f is an increasing function.

24. Find $f(n)$ when $n = 2^k$, where f satisfies the recurrence relation $f(n) = 8f(n/2) + n^2$ with $f(1) = 1$.

25. Estimate the size of f in Exercise 24 if f is an increasing function.

5.4

Generating Functions

INTRODUCTION

Generating functions are used to represent sequences efficiently by coding the terms of a sequence as coefficients of powers of a variable x in a formal power series. Generating functions can be used to solve many types of counting problems, such as the number of ways to select or distribute objects of different kinds, subject to a variety of constraints, and the number of ways to make change for a dollar using coins of different denominations. Generating functions can be used to solve recurrence relations by translating a recurrence relation for the terms of a sequence into an equation involving a generating function. This equation then can be solved to find a closed form for the generating function. From this closed form, the coefficients of the power series for the generating function can be found, solving the original recurrence relation. Generating functions can also be used to prove combinatorial identities by taking advantage of relatively simple relationships between functions that can be translated into identities involving the terms of sequences. Generating functions are a helpful tool for studying many properties of sequences beside those described in this section, such as their use for establishing asymptotic formulae for the terms of a sequence.

We begin with the definition of the generating function for a sequence.

DEFINITION 1. The *generating function* for the sequence $a_0, a_1, \ldots, a_k, \ldots$ of real numbers is the infinite series

$$G(x) = a_0 + a_1 x + \cdots + a_k x^k + \cdots = \sum_{k=0}^{\infty} a_k x^k.$$

Remark: The generating function for $\{a_k\}$ given in Definition 1 is sometimes called the **ordinary generating function** of $\{a_k\}$ to distinguish it from other types of generating functions for this sequence.

EXAMPLE 1 The generating functions for the sequences $\{a_k\}$ with $a_k = 3$, $a_k = k+1$, and $a_k = 2^k$ are $\sum_{k=0}^{\infty} 3x^k$, $\sum_{k=0}^{\infty}(k+1)x^k$, and $\sum_{k=0}^{\infty} 2^k x^k$, respectively. ■

We can define generating functions for finite sequences of real numbers by extending a finite sequence $a_0, a_1, \ldots, a_n$ into an infinite sequence by setting $a_{n+1} = 0$, $a_{n+2} = 0$, and so on. The generating function $G(x)$ of this infinite sequence $\{a_n\}$ is a polynomial of degree n since no terms of the form $a_j x^j$ with $j > n$ occur, that is,

$$G(x) = a_0 + a_1 x + \cdots + a_n x^n.$$

EXAMPLE 2 What is the generating function for the sequence 1, 1, 1, 1, 1, 1?

Solution: The generating function of 1, 1, 1, 1, 1, 1 is

$$1 + x + x^2 + x^3 + x^4 + x^5.$$

By Example 6 of Section 3.2 we have

$$(x^6 - 1)/(x - 1) = 1 + x + x^2 + x^3 + x^4 + x^5.$$

Consequently, $G(x) = (x^6 - 1)/(x - 1)$ is the generating function of the sequence 1, 1, 1, 1, 1, 1. ■

EXAMPLE 3 Let m be a positive integer. Let $a_k = C(m, k)$, for $k = 0, 1, 2, \ldots, m$. What is the generating function for the sequence $a_0, a_1, \ldots, a_m$?

Solution: The generating function for this sequence is

$$G(x) = C(m, 0) + C(m, 1)x + C(m, 2)x^2 + \cdots + C(m, m)x^m.$$

The binomial theorem shows that $G(x) = (1 + x)^m$. ■

USEFUL FACTS ABOUT POWER SERIES

When generating functions are used to solve counting problems, they are usually considered to be **formal power series**. Questions about the convergence of these series are ignored. However, to apply some results from calculus, it is sometimes important to consider for which x the power series converges. We will not be concerned with questions of convergence in our discussions. Readers familiar with calculus can consult textbooks on this subject for details about the convergence of the series we consider here.

We will state now some important facts about infinite series used when working with generating functions. A discussion of these and related results can be found in calculus texts.

EXAMPLE 4 The function $f(x) = 1/(1 - x)$ is the generating function of the sequence 1, 1, 1, 1, ..., since

$$1/(1 - x) = 1 + x + x^2 + \cdots$$

for $|x| < 1$. ■

EXAMPLE 5 The function $f(x) = 1/(1-ax)$ is the generating function of the sequence $1, a, a^2, a^3, \ldots$, since

$$1/(1-ax) = 1 + ax + a^2x^2 + \cdots$$

when $|ax| < 1$, or equivalently, for $|x| < 1/|a|$ for $a \neq 0$. ■

We also will need some results on how to add and how to multiply two generating functions. Proofs of these results can be found in calculus texts.

THEOREM 1 Let $f(x) = \sum_{k=0}^{\infty} a_k x^k$ and $g(x) = \sum_{k=0}^{\infty} b_k x^k$. Then

$$f(x) + g(x) = \sum_{k=0}^{\infty} (a_k + b_k)x^k \quad \text{and} \quad f(x)g(x) = \sum_{k=0}^{\infty} \left(\sum_{j=0}^{k} a_j b_{k-j} \right) x^k.$$

We will illustrate how Theorem 1 can be used with the following example.

EXAMPLE 6 Let $f(x) = 1/(1-x)^2$. Use Example 4 to find the coefficients $a_0, a_1, a_2, \ldots$ in the expansion $f(x) = \sum_{k=0}^{\infty} a_k x^k$.

Solution: From Example 4 we see that

$$1/(1-x) = 1 + x + x^2 + x^3 + \cdots.$$

Hence, from Theorem 1, we have

$$1/(1-x)^2 = \sum_{k=0}^{\infty} \left(\sum_{j=0}^{k} 1 \right) x^k = \sum_{k=0}^{\infty} (k+1)x^k.$$

■

Remark: This result also can be derived from Example 4 by differentiation. Taking derivatives is a useful technique for producing new identities from existing identities for generating functions.

To use generating functions to solve many important counting problems, we will need to apply the binomial theorem for exponents that are not positive integers. Before we state an extended version of the binomial theorem, we need to define extended binomial coefficients.

DEFINITION 2. Let u be a real number and k a nonnegative integer. Then the *extended binomial coefficient* $\binom{u}{k}$ is defined by

$$\binom{u}{k} = \begin{cases} u(u-1)\cdots(u-k+1)/k! & \text{if } k > 0, \\ 1 & \text{if } k = 0. \end{cases}$$

EXAMPLE 7 Find the values of the extended binomial coefficients $\binom{-2}{3}$ and $\binom{1/2}{3}$.

Solution: Taking $u = -2$ and $k = 3$ in Definition 2 gives us

$$\binom{-2}{3} = \frac{(-2)(-3)(-4)}{3!} = -4.$$

Similarly, taking $u = 1/2$ and $k = 3$ gives us

$$\begin{aligned}\binom{1/2}{3} &= \frac{(1/2)(1/2-1)(1/2-2)}{3!}\\ &= (1/2)(-1/2)(-3/2)/6\\ &= 1/16.\end{aligned}$$

■

The following example provides a useful formula for extended binomial coefficients when the top parameter is a negative integer. It will be useful in our subsequent discussions.

EXAMPLE 8 When the top parameter is a negative integer, the extended binomial coefficient can be expressed in terms of an ordinary binomial coefficient. To see that this is the case, note that

$$\begin{aligned}\binom{-n}{r} &= \frac{(-n)(-n-1)\cdots(-n-r+1)}{r!}\\ &= \frac{(-1)^r n(n+1)\cdots(n+r-1)}{r!}\\ &= \frac{(-1)^r(n+r-1)(n+r-2)\cdots n}{r!}\\ &= \frac{(-1)^r(n+r-1)!}{r!(n-1)!}\\ &= (-1)^r\binom{n+r-1}{r}\\ &= (-1)^r C(n+r-1,r).\end{aligned}$$

■

We now state the extended binomial theorem.

THEOREM 2 **THE EXTENDED BINOMIAL THEOREM** **Let x be a real number with $|x| < 1$ and let u be a real number. Then**

$$(1+x)^u = \sum_{k=0}^{\infty}\binom{u}{k}x^k.$$

Theorem 2 can be proved using the theory of the Maclaurin series. We leave its proof to the reader with a familiarity with this part of calculus.

Remark: When u is a positive integer, the extended binomial theorem reduces to the binomial theorem presented in Section 4.3, since in that case $\binom{u}{k} = 0$ if $k > u$.

The following example illustrates the use of Theorem 2 when the exponent is a negative integer.

EXAMPLE 9 Find the generating functions for $(1+x)^{-n}$ and $(1-x)^{-n}$ where n is a positive integer, using the extended binomial theorem.

Solution: By the extended binomial theorem, it follows that

$$(1+x)^{-n} = \sum_{k=0}^{\infty} \binom{-n}{k} x^k.$$

Using Example 8, which provides a simple formula for $\binom{-n}{k}$, we obtain

$$(1+x)^{-n} = \sum_{k=0}^{\infty} (-1)^k C(n+k-1, k) x^k.$$

Replacing x by $-x$, we find that

$$(1-x)^{-n} = \sum_{k=0}^{\infty} C(n+k-1, k) x^k.$$

■

Table 1 presents a useful summary of some generating functions that arise frequently.

COUNTING PROBLEMS AND GENERATING FUNCTIONS

Generating functions can be used to solve a wide variety of counting problems. In particular, they can be used to count the number of combinations of various types. In Chapter 4 we developed techniques to count the r-combinations from a set with n elements when repetition is allowed and additional constraints may exist. Such problems are equivalent to counting the solutions to equations of the form

$$e_1 + e_2 + \cdots + e_n = C,$$

where C is a constant and each e_i is a nonnegative integer that may be subject to a specified constraint. Generating functions can also be used to solve counting problems of this type, as the following examples show.

EXAMPLE 10 Find the number of solutions of

$$e_1 + e_2 + e_3 = 17,$$

where e_1, e_2, and e_3 are nonnegative integers with $2 \le e_1 \le 5$, $3 \le e_2 \le 6$, and $4 \le e_3 \le 7$.

Solution: The number of solutions with the indicated constraints is the coefficient of x^{17} in the expansion of

$$(x^2 + x^3 + x^4 + x^5)(x^3 + x^4 + x^5 + x^6)(x^4 + x^5 + x^6 + x^7).$$

TABLE 1 Useful Generating Functions.

$G(x)$	a_k
$(1+x)^n = \sum_{k=0}^{n} C(n,k)x^k$ $= 1 + C(n,1)x + C(n,2)x^2 + \cdots + x^n$	$C(n,k)$
$(1+ax)^n = \sum_{k=0}^{n} C(n,k)a^k x^k$ $= 1 + C(n,1)ax + C(n,2)a^2x^2 + \cdots + a^n x^n$	$C(n,k)a^k$
$(1+x^r)^n = \sum_{k=0}^{n} C(n,k)x^{rk}$ $= 1 + C(n,1)x^r + C(n,2)x^{2r} + \cdots + x^{rn}$	$C(n,k/r)$ if $r \mid k$; 0 otherwise
$\frac{1-x^{n+1}}{1-x} = \sum_{k=0}^{n} x^k = 1 + x + x^2 + \cdots + x^n$	1 if $k \le n$; 0 otherwise
$\frac{1}{1-x} = \sum_{k=0}^{\infty} x^k = 1 + x + x^2 + \cdots$	1
$\frac{1}{1-ax} = \sum_{k=0}^{\infty} a^k x^k = 1 + ax + a^2x^2 + \cdots$	a^k
$\frac{1}{1-x^r} = \sum_{k=0}^{\infty} x^{rk} = 1 + x^r + x^{2r} + \cdots$	1 if $r \mid k$; 0 otherwise
$\frac{1}{(1-x)^2} = \sum_{k=0}^{\infty}(k+1)x^k = 1 + 2x + 3x^2 + \cdots$	$k+1$
$\frac{1}{(1-x)^n} = \sum_{k=0}^{\infty} C(n+k-1,k)x^k$ $= 1 + C(n,1)x + C(n+1,2)x^2 + \cdots$	$C(n+k-1,k) = C(n+k-1,n-1)$
$\frac{1}{(1+x)^n} = \sum_{k=0}^{\infty} C(n+k-1,k)(-1)^k x^k$ $= 1 - C(n,1)x + C(n+1,2)x^2$ $- C(n+2,3)x^3 + \cdots$	$(-1)^k C(n+k-1,k) = (-1)^k C(n+k-1,n-1)$
$\frac{1}{(1-ax)^n} = \sum_{k=0}^{\infty} C(n+k-1,k)a^k x^k$ $= 1 + C(n,1)ax + C(n+1,2)a^2x^2 + \cdots$	$C(n+k-1,k)a^k = C(n+k-1,n-1)a^k$
$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$	$1/k!$
$\ln(1+x) = \sum_{k=0}^{\infty} \frac{(-1)^{k+1}}{k} x^k$ $= x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \cdots$	$(-1)^{k+1}/k$

Note: The series for the last two generating functions can be found in most calculus books when power series are discussed.

This follows since we obtain a term equal to x^{17} in the product by picking a term in the first sum x^{e_1}, a term in the second sum x^{e_2}, and a term in the third sum x^{e_3}, where the exponents e_1, e_2, and e_3 satisfy the equation $e_1 + e_2 + e_3 = 17$ and the given constraints.

It is not hard to see that the coefficient of x^{17} in this product is 3. Hence, there are three solutions. (Note that the calculating of this coefficient involves about as much work as enumerating all the solutions of the equation with the given constraints. However, the method that this illustrates often can be used to solve wide classes of counting problems with special formulae, as we will see. Furthermore, a computer algebra system can be used to do such computations.) ■

EXAMPLE 11 In how many different ways can eight identical cookies be distributed among three distinct children if each child receives at least two cookies and no more than four cookies?

Solution: Since each child receives at least two but no more than four cookies, for each child there is a factor equal to

$$(x^2 + x^3 + x^4)$$

in the generating function for the sequence $\{c_n\}$, where c_n is the number of ways to distribute n cookies. Since there are three children, this generating function is

$$(x^2 + x^3 + x^4)^3.$$

We need the coefficient of x^8 in this product. The reason is that the x^8 terms in the expansion correspond to the ways that three terms can be selected, with one from each factor, that have exponents adding up to 8. Furthermore, the exponents of the term from the first, second, and third factors are the numbers of cookies the first, second, and third children receive, respectively. Computation shows that this coefficient equals 6. Hence, there are six ways to distribute the cookies so that each child receives at least two, but no more than four, cookies. ■

EXAMPLE 12 Use generating functions to determine the number of ways to insert tokens worth \$1, \$2, and \$5 into a vending machine to pay for an item that costs r dollars in both the cases when the order in which the tokens are inserted does not matter and when the order does matter. (For example, there are two ways to pay for an item that costs \$3 when the order in which the tokens are inserted does not matter: inserting three \$1 tokens or one \$1 token and a \$2 token. When the order matters, there are three ways: inserting three \$1 tokens, inserting a \$1 token and then a \$2 token, or inserting a \$2 token and then a \$1 token.)

Solution: Consider the case when the order in which the tokens are inserted does not matter. Here, all we care about is the number of each token used to produce a total of r dollars. Since we can use any number of \$1 tokens, any number of \$2 tokens, and any number of \$5 tokens, the answer is the coefficient of x^r in the generating function

$$(1 + x + x^2 + x^3 + \cdots)(1 + x^2 + x^4 + x^6 + \cdots)(1 + x^5 + x^{10} + x^{15} + \cdots).$$

(The first factor in this product represents the \$1 tokens used, the second the \$2 tokens used, and the third the \$5 tokens used.) For example, the number of ways to pay for an item costing \$7 using \$1, \$2, and \$5 tokens is given by the coefficient of x^7 in this expansion, which equals 6.

When the order in which the tokens are inserted matters, the number of ways to insert exactly n tokens to produce a total of r dollars is the coefficient of x^r in

$$(x + x^2 + x^5)^n,$$

since each of the r tokens may be a \$1 token, a \$2 token, or a \$5 token. Since any number of tokens may be inserted, the number of ways to produce r dollars using \$1, \$2, or \$5 tokens, when the order in which the tokens are inserted matters, is the coefficient of x^r in

$$\begin{aligned} 1 + (x + x^2 + x^5) + (x + x^2 + x^5)^2 + \cdots &= \frac{1}{1 - (x + x^2 + x^5)} \\ &= \frac{1}{1 - x - x^2 - x^5}, \end{aligned}$$

where we have added the number of ways to insert 0 tokens, 1 token, 2 tokens, 3 tokens, and so on, and where we have used the identity $1/(1 - x) = 1 + x + x^2 + \cdots$ with x replaced with $x + x^2 + x^5$. For example, the number of ways to pay for an item costing \$7 using \$1, \$2, and \$5 tokens, when the order in which the tokens are used matters, is the coefficient of x^7 in this expansion, which equals 26. [To see that this coefficient equals 26 requires the addition of the coefficients of x^7 in the expansions $(x + x^2 + x^5)^k$ for $2 \le k \le 7$. This can be done by hand with considerable computation, or a computer algebra system can be used.] ■

The following example shows the versatility of generating functions when used to solve problems with differing assumptions.

EXAMPLE 13 Use generating functions to find the number of k-combinations of a set with n elements. Assume that the binomial theorem has already been established.

Solution: Each of the n elements in the set contributes the term $(1 + x)$ to the generating function $f(x) = \sum_{k=0}^{n} a_k x^k$. Here $f(x)$ is the generating function for $\{a_k\}$, where a_k represents the number of k-combinations of a set with n elements. Hence,

$$f(x) = (1 + x)^n.$$

But by the binomial theorem, we have

$$f(x) = \sum_{k=0}^{n} \binom{n}{k} x^k,$$

where

$$\binom{n}{k} = \frac{n!}{k!(n - k)!}.$$

Hence, $C(n, k)$, the number of k-combinations of a set with n elements, is

$$\frac{n!}{k!(n - k)!}.$$

■

Remark: We proved the binomial theorem in Section 4.3 using the formula for the number of r-combinations of a set with n elements. This example shows that the binomial theorem, which can be proved by mathematical induction, can be used to derive the formula for the number of r-combinations of a set with n elements.

EXAMPLE 14 Use generating functions to find the number of r-combinations from a set with n elements when repetition of elements is allowed.

Solution: Let $G(x)$ be the generating function for the sequence $\{a_r\}$, where a_r equals the number of r-combinations of a set with n elements with repetitions allowed. That is, $G(x) = \sum_{r=0}^{\infty} a_r x^r$. Since we can select any number of a particular member of the set with n elements when we form an r-combination with repetition allowed, each of the n elements contributes $(1 + x + x^2 + x^3 + \cdots)$ to a product expansion for $G(x)$. Each element contributes this factor since it may be selected zero times, one time, two times, three times, and so on, when an r-combination is formed (with a total of r elements selected). Since there are n elements in the set and each contributes this same factor to $G(x)$, we have

$$G(x) = (1 + x + x^2 + \cdots)^n.$$

As long as $|x| < 1$, we have $1 + x + x^2 + \cdots = 1/(1 - x)$, so

$$G(x) = 1/(1 - x)^n = (1 - x)^{-n}.$$

Applying the extended binomial theorem, it follows that

$$(1 - x)^{-n} = (1 + (-x))^{-n} = \sum_{r=0}^{\infty} \binom{-n}{r} (-x)^r.$$

The number of r-combinations of a set with n elements with repetitions allowed, when r a positive integer, is the coefficient a_r of x^r in this sum. Consequently, using Example 8 we find that a_r equals

$$\begin{aligned}\binom{-n}{r}(-1)^r &= (-1)^r C(n + r - 1, r) \cdot (-1)^r \\ &= C(n + r - 1, r).\end{aligned}$$

■

Note that this is the same result we stated as Theorem 2 in Section 4.6.

EXAMPLE 15 Use generating functions to find the number of ways to select r objects of n different kinds if we must select at least one object of each kind.

Solution: Since we need to select at least one object of each kind, each of the n kinds of objects contributes the factor $(x + x^2 + x^3 + \cdots)$ to the generating function $G(x)$ for the sequence $\{a_r\}$, where a_r is the number of ways to select r objects of n different kinds if we need at least one object of each kind. Hence,

$$G(x) = (x + x^2 + x^3 + \cdots)^n = x^n (1 + x + x^2 + \cdots)^n = x^n/(1 - x)^n.$$

Using the extended binomial theorem and Example 8, we have

$$\begin{aligned}G(x) &= x^n/(1-x)^n \\ &= x^n \cdot (1-x)^{-n} \\ &= x^n \sum_{r=0}^{\infty} \binom{-n}{r}(-x)^r \\ &= x^n \sum_{r=0}^{\infty} (-1)^r C(n+r-1, r)(-1)^r x^r \\ &= \sum_{r=0}^{\infty} C(n+r-1, r) x^{n+r} \\ &= \sum_{t=n}^{\infty} C(t-1, t-n) x^t \\ &= \sum_{r=n}^{\infty} C(r-1, r-n) x^r.\end{aligned}$$

We have shifted the summation in the next-to-last equality by setting $t = n + r$ so that $t = n$ when $r = 0$ and $n + r - 1 = t - 1$, and then we replaced t by r as the index of summation in the last equality to return to our original notation. Hence, there are $C(r-1, r-n)$ ways to select r objects of n different kinds if we must select at least one object of each kind. ■

USING GENERATING FUNCTIONS TO SOLVE RECURRENCE RELATIONS

We can find the solution to a recurrence relation and its initial conditions by finding an explicit formula for the associated generating function. This is illustrated in the following examples.

EXAMPLE 16 Solve the recurrence relation $a_k = 3a_{k-1}$ for $k = 1, 2, 3, \ldots$ and initial condition $a_0 = 2$.

Solution: Let $G(x)$ be the generating function for the sequence $\{a_k\}$, that is, $G(x) = \sum_{k=0}^{\infty} a_k x^k$. First note that

$$xG(x) = \sum_{k=0}^{\infty} a_k x^{k+1} = \sum_{k=1}^{\infty} a_{k-1} x^k.$$

Using the recurrence relation, we see that

$$\begin{aligned}G(x) - 3xG(x) &= \sum_{k=0}^{\infty} a_k x^k - 3\sum_{k=1}^{\infty} a_{k-1} x^k \\ &= a_0 + \sum_{k=1}^{\infty} (a_k - 3a_{k-1}) x^k \\ &= 2,\end{aligned}$$

since $a_0 = 2$ and $a_k = 3a_{k-1}$. Thus,

$$G(x) - 3xG(x) = (1 - 3x)G(x) = 2.$$

Solving for $G(x)$ shows that $G(x) = 2/(1 - 3x)$. Using the identity $1/(1 - ax) = \sum_{k=0}^{\infty} a^k x^k$, from Table 1, we have

$$G(x) = 2\sum_{k=0}^{\infty} 3^k x^k = \sum_{k=0}^{\infty} 2 \cdot 3^k x^k.$$

Consequently, $a_k = 2 \cdot 3^k$. ■

EXAMPLE 17 Suppose that a valid code word is an n-digit number in decimal notation containing an even number of 0s. Let a_n denote the number of valid code words of length n. In Example 7 of Section 5.1 we showed that the sequence $\{a_n\}$ satisfies the recurrence relation

$$a_n = 8a_{n-1} + 10^{n-1}$$

and the initial condition $a_1 = 9$. Use generating functions to find an explicit formula for a_n.

Solution: To make our work with generating functions simpler, we extend this sequence by setting $a_0 = 1$; when we assign this value to a_0 and use the recurrence relation, we have $a_1 = 8a_0 + 10^0 = 8 + 1 = 9$, which is consistent with our original initial condition. (It also makes sense because there is one code word of length 0—the empty string.)

We multiply both sides of the recurrence relation by x^n to obtain

$$a_n x^n = 8a_{n-1}x^n + 10^{n-1}x^n.$$

Let $G(x) = \sum_{n=0}^{\infty} a_n x^n$ be the generating function of the sequence $a_0, a_1, a_2, \ldots$. We sum both sides of the last equation starting with $n = 1$, to find that

$$\begin{aligned}
G(x) - 1 = \sum_{n=1}^{\infty} a_n x^n &= \sum_{n=1}^{\infty} (8a_{n-1}x^n + 10^{n-1}x^n) \\
&= 8\sum_{n=1}^{\infty} a_{n-1}x^n + \sum_{n=1}^{\infty} 10^{n-1}x^n \\
&= 8x\sum_{n=1}^{\infty} a_{n-1}x^{n-1} + x\sum_{n=1}^{\infty} 10^{n-1}x^{n-1} \\
&= 8x\sum_{n=0}^{\infty} a_n x^n + x\sum_{n=0}^{\infty} 10^n x^n \\
&= 8xG(x) + x/(1 - 10x),
\end{aligned}$$

where we have used Example 5 to evaluate the second summation. Therefore, we have

$$G(x) - 1 = 8xG(x) + x/(1 - 10x).$$

Solving for $G(x)$ shows that

$$G(x) = \frac{1 - 9x}{(1 - 8x)(1 - 10x)}.$$

Expanding the right-hand side of this equation into partial fractions (as is done in the integration of rational functions studied in calculus) gives

$$G(x) = \frac{1}{2}\left(\frac{1}{1-8x} + \frac{1}{1-10x}\right).$$

Using Example 5 twice (once with $a = 8$ and once with $a = 10$) gives

$$\begin{aligned} G(x) &= \frac{1}{2}\left(\sum_{n=0}^{\infty} 8^n x^n + \sum_{n=0}^{\infty} 10^n x^n\right) \\ &= \sum_{n=0}^{\infty} \frac{1}{2}(8^n + 10^n). \end{aligned}$$

Consequently, we have shown that

$$a_n = \frac{1}{2}(8^n + 10^n).$$

■

USING GENERATING FUNCTIONS TO PROVE IDENTITIES

In Chapter 4 we saw how combinatorial identities could be established using combinatorial proofs. Here we will show that such identities, as well as identities for extended binomial coefficients, can be proved using generating functions. Sometimes the generating function approach is simpler than other approaches, especially when it is simpler to work with the closed form of a generating function than with the terms of the sequence themselves. We illustrate how generating functions can be used to prove identities with the following example.

EXAMPLE 18 Use generating functions to show that

$$\sum_{k=0}^{n} C(n,k)^2 = C(2n,n)$$

whenever n is a positive integer.

Solution: First note that by the binomial theorem $C(2n, n)$ is the coefficient of x^n in $(1+x)^{2n}$. However, we also have

$$\begin{aligned} (1+x)^{2n} &= [(1+x)^n]^2 \\ &= [C(n,0) + C(n,1)x + C(n,2)x^2 + \cdots + C(n,n)x^n]^2. \end{aligned}$$

The coefficient of x^n in this expression is $C(n,0)C(n,n) + C(n,1)C(n,n-1) + C(n,2)C(n,n-2) + \ldots + C(n,n)C(n,0)$. This equals $\sum_{k=0}^{n} C(n,k)^2$, since $C(n,n-k) = C(n,k)$. Since both $C(2n,n)$ and $\sum_{k=0}^{n} C(n,k)^2$ represent the coefficient of x^n in $(1+x)^{2n}$, they must be equal. ■

Exercises 42 and 43 at the end of this section ask that Pascal's identity and Vandermonde's identity be proved using generating functions.

Exercises

1. Find the generating function for the finite sequence 2, 2, 2, 2, 2, 2.

2. Find the generating function for the finite sequence 1, 4, 16, 64, 256.

3. Find a closed form for the generating function for each of the following sequences. (Assume a general form for the terms of the sequence, using the most obvious choice of such a sequence.)

a) $0, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, \ldots$

b) $0, 0, 0, 1, 1, 1, 1, 1, 1, \ldots$

c) $0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, \ldots$

d) $2, 4, 8, 16, 32, 64, 128, 256, \ldots$

e) $\binom{7}{0}, \binom{7}{1}, \binom{7}{2}, \ldots, \binom{7}{7}, 0, 0, 0, 0, 0, \ldots$

f) $2, -2, 2, -2, 2, -2, 2, -2, \ldots$

g) $1, 1, 0, 1, 1, 1, 1, 1, 1, 1, \ldots$

h) $0, 0, 0, 1, 2, 3, 4, \ldots$

4. Find a closed form for the generating function for each of the following sequences. (Assume a general form for the terms of the sequence, using the most obvious choice of such a sequence.)

a) $-1, -1, -1, -1, -1, -1, -1, 0, 0, 0, 0, 0, 0, \ldots$

b) $1, 3, 9, 27, 81, 243, 729, \ldots$

c) $0, 0, 3, -3, 3, -3, 3, -3, \ldots$

d) $1, 2, 1, 1, 1, 1, 1, 1, 1, \ldots$

e) $\binom{7}{0}, 2\binom{7}{1}, 2^2\binom{7}{2}, \ldots, 2^7\binom{7}{7}, 0, 0, 0, 0, \ldots$

f) $-3, 3, -3, 3, -3, 3, \ldots$

g) $0, 1, -2, 4, -8, 16, -32, 64, \ldots$

h) $1, 0, 1, 0, 1, 0, 1, 0, \ldots$

5. Find a closed form for the generating function for the sequence $\{a_n\}$ where

a) $a_n = 5$ for all $n = 0, 1, 2, \ldots$.

b) $a_n = 3^n$ for all $n = 0, 1, 2, \ldots$.

c) $a_n = 2$ for $n = 3, 4, 5, \ldots$ and $a_0 = a_1 = a_2 = 0$.

d) $a_n = 2n + 3$ for all $n = 0, 1, 2, \ldots$.

e) $a_n = \binom{8}{n}$ for all $n = 0, 1, 2, \ldots$.

f) $a_n = \binom{n+4}{n}$ for all $n = 0, 1, 2, \ldots$.

6. Find a closed form for the generating function for the sequence $\{a_n\}$ where

a) $a_n = -1$ for all $n = 0, 1, 2, \ldots$.

b) $a_n = 2^n$ for $n = 1, 2, 3, 4, \ldots$ and $a_0 = 0$.

c) $a_n = n - 1$ for $n = 0, 1, 2, \ldots$.

d) $a_n = 1/(n + 1)!$ for $n = 0, 1, 2, \ldots$.

e) $a_n = \binom{n}{2}$ for $n = 0, 1, 2, \ldots$.

f) $a_n = \binom{10}{n+1}$ for $n = 0, 1, 2, \ldots$.

7. For each of the following generating functions, provide a closed formula for the sequence it determines.

a) $(3x - 4)^3$

b) $(x^3 + 1)^3$

c) $1/(1 - 5x)$

d) $x^3/(1 + 3x)$

e) $x^2 + 3x + 7 + (1/(1 - x^2))$

f) $(x^4/(1 - x^4)) - x^3 - x^2 - x - 1$

g) $x^2/(1 - x)^2$

h) $2e^{2x}$

8. For each of the following generating functions, provide a closed formula for the sequence it determines.

a) $(x^2 + 1)^3$

b) $(3x - 1)^3$

c) $1/(1 - 2x^2)$

d) $x^2/(1 - x)^3$

e) $x - 1 + (1/(1 - 3x))$

f) $(1 + x^3)/(1 + x)^3$

***g)** $x/(1 + x + x^2)$

h) $e^{3x^2} - 1$

9. Find the coefficient of x^{10} in the power series of each of the following functions.

a) $(1 + x^5 + x^{10} + x^{15} + \cdots)^3$

b) $(x^3 + x^4 + x^5 + x^6 + x^7 + \cdots)^3$

c) $(x^4 + x^5 + x^6)(x^3 + x^4 + x^5 + x^6 + x^7)(1 + x + x^2 + x^3 + x^4 + \cdots)$

d) $(x^2 + x^4 + x^6 + x^8 + \cdots)(x^3 + x^6 + x^9 + \cdots)(x^4 + x^8 + x^{12} + \cdots)$

e) $(1 + x^2 + x^4 + x^6 + x^8 + \cdots)(1 + x^4 + x^8 + x^{12} + \cdots)(1 + x^6 + x^{12} + x^{18} + \cdots)$

10. Find the coefficient of x^9 in the power series of each of the following functions.

a) $(1 + x^3 + x^6 + x^9 + \cdots)^3$

b) $(x^2 + x^3 + x^4 + x^5 + x^6 + \cdots)^3$

c) $(x^3 + x^5 + x^6)(x^3 + x^4)(x + x^2 + x^3 + x^4 + \cdots)$

d) $(x + x^4 + x^7 + x^{10} + \cdots)(x^2 + x^4 + x^6 + x^8 + \cdots)$

e) $(1 + x + x^2)^3$

11. Find the coefficient of x^{10} in the power series of each of the following functions.

a) $1/(1 - 2x)$

b) $1/(1 + x)^2$

c) $1/(1 - x)^3$

d) $1/(1 + 2x)^4$

e) $x^4/(1 - 3x)^3$

12. Find the coefficient of x^{12} in the power series of each of the following functions.

a) $1/(1 + 3x)$

b) $1/(1 - 2x)^2$

c) $1/(1 + x)^8$

d) $1/(1 - 4x)^3$

e) $x^3/(1 + 4x)^2$

13. Use generating functions to determine the number of different ways 10 identical balloons can be given to four children if each child receives at least two balloons.

14. Use generating functions to determine the number of different ways 12 identical action figures can be given to five children so that each child receives at most three action figures.

15. Use generating functions to determine the number of different ways 15 identical stuffed animals can be given to six children so that each child receives at least one but no more than three stuffed animals.

16. Use generating functions to find the number of ways to choose a dozen bagels from three varieties—egg, salty, and plain—if at least two bagels of each kind but no more than three salty bagels are chosen.

17. In how many ways can 25 identical donuts be distributed to four police officers so that each officer gets at least three but no more than seven donuts?

18. Use generating functions to find the number of ways to select 14 balls from a jar containing 100 red balls, 100 blue balls, and 100 green balls so that no fewer than 3 and no more than 10 blue balls are selected. Assume that the order in which the balls are drawn does not matter.

19. What is the generating function for the sequence $\{c_k\}$, where c_k is the number of ways to make change for k dollars using \$1 bills, \$2 bills, \$5 bills, and \$10 bills?

20. What is the generating function for the sequence $\{c_k\}$, where c_k represents the number of ways to make change for k pesos using bills worth 10 pesos, 20 pesos, 50 pesos, and 100 pesos?

21. Give a combinatorial interpretation of the coefficient of x^4 in the expansion $(1 + x + x^2 + x^3 + \cdots)^3$. Use this interpretation to find this number.

22. Give a combinatorial interpretation of the coefficient of x^6 in the expansion $(1 + x + x^2 + x^3 + \cdots)^n$. Use this interpretation to find this number.

23. **a)** What is the generating function for $\{a_k\}$, where a_k is the number of solutions of $x_1 + x_2 + x_3 = k$ when x_1, x_2, and x_3 are integers with $x_1 \geq 2$, $0 \leq x_2 \leq 3$, and $2 \leq x_3 \leq 5$?

b) Use your answer to part (a) to find a_6.

24. **a)** What is the generating function for $\{a_k\}$, where a_k is the number of solutions of $x_1 + x_2 + x_3 + x_4 = k$ when x_1, x_2, x_3, and x_4 are integers with $x_1 \geq 3$, $1 \leq x_2 \leq 5$, $0 \leq x_3 \leq 4$, and $x_4 \geq 1$?

b) Use your answer to part (a) to find a_7.

25. Explain how generating functions can be used to find the number of ways in which postage of r cents can be pasted on an envelope using 3-cent, 4-cent, and 20-cent stamps.

a) Assume that the order the stamps are pasted on does not matter.

b) Assume that the stamps are pasted in a row and the order in which they are pasted on matters.

c) Use your answer to part (a) to determine the number of ways 46 cents of postage can be pasted on an envelope using 3-cent, 4-cent, and 20-cent stamps when the order the stamps are pasted on does not matter. (Use of a computer algebra program is advised.)

d) Use your answer to part (b) to determine the number of ways 46 cents of postage can be pasted in a row on an envelope using 3-cent, 4-cent, and 20-cent stamps when the order in which the stamps are pasted on matters. (Use of a computer algebra program is advised.)

26. **a)** Show that $1/(1 - x - x^2 - x^3 - x^4 - x^5 - x^6)$ is the generating function for the number of ways that the sum n can be obtained when a die is rolled repeatedly and the order of the rolls matters.

b) Use part (a) to find the number of ways to roll a total of 8 when a die is rolled repeatedly, and the order of the rolls matters. (Use of a computer algebra package is advised.)

27. Use generating functions (and a computer algebra package, if available) to find the number of ways to make change for \$1 using

a) dimes and quarters.

b) nickels, dimes, and quarters.

c) pennies, dimes, and quarters.

d) pennies, nickels, dimes, and quarters.

28. Use generating functions (and a computer algebra package, if available) to find the number of ways to make change for \$1 using pennies, nickels, dimes, and quarters with

a) no more than 10 pennies.

b) no more than 10 pennies and no more than 10 nickels.

***c)** no more than 10 coins.

29. Use generating functions to find the number of ways to make change for \$100 using

a) \$10, \$20, and \$50 bills.

b) \$5, \$10, \$20, and \$50 bills.

c) \$5, \$10, \$20, and \$50 bills if at least one bill of each denomination is used.

d) \$5, \$10, and \$20 bills if at least one and no more than four of each denomination is used.

30. If $G(x)$ is the generating function for the sequence $\{a_k\}$, what is the generating function for each of the following sequences?

a) $2a_0, 2a_1, 2a_2, 2a_3, \ldots$ **b)** $0, a_0, a_1, a_2, a_3, \ldots$

c) $0, 0, 0, 0, a_2, a_3, \ldots$ **d)** $a_2, a_3, a_4, \ldots$

e) $a_1, 2a_2, 3a_3, 4a_4, \ldots$ (*Hint:* Calculus is required here.)

f) $a_0^2, 2a_0a_1, a_1^2 + 2a_0a_2, 2a_0a_3 + 2a_1a_2, 2a_0a_4 + 2a_1a_3 + a_2^2, \ldots$

31. If $G(x)$ is the generating function for the sequence $\{a_k\}$, what is the generating function for each of the following sequences?

- **a)** $0, 0, 0, a_3, a_4, a_5, \ldots$
- **b)** $a_0, 0, a_1, 0, a_2, 0, \ldots$
- **c)** $0, 0, 0, 0, a_0, a_1, a_2, \ldots$
- **d)** $a_0, 2a_1, 4a_2, 8a_3, 16a_4, \ldots$
- **e)** $a_0, a_1/2, a_2/3, a_3/4, \ldots$ (*Hint:* Calculus is required here.)
- **f)** $a_0, a_0 + a_1, a_0 + a_1 + a_2, a_0 + a_1 + a_2 + a_3, \ldots$

32. Use generating functions to solve the recurrence relation $a_k = 7a_{k-1}$ with the initial condition $a_0 = 5$.

33. Use generating functions to solve the recurrence relation $a_k = 3a_{k-1} + 2$ with the initial condition $a_0 = 1$.

34. Use generating functions to solve the recurrence relation $a_k = 3a_{k-1} + 4^{k-1}$ with the initial condition $a_0 = 1$.

35. Use generating functions to solve the recurrence relation $a_k = 5a_{k-1} - 6a_{k-2}$ with initial conditions $a_0 = 6$ and $a_1 = 30$.

36. Use generating functions to solve the recurrence relation $a_k = a_{k-1} + 2a_{k-2} + 2^k$ with initial conditions $a_0 = 4$ and $a_1 = 12$.

37. Use generating functions to solve the recurrence relation $a_k = 4a_{k-1} - 4a_{k-2} + k^2$ with initial conditions $a_0 = 2$ and $a_1 = 5$.

38. Use generating functions to solve the recurrence relation $a_k = 2a_{k-1} + 3a_{k-2} + 4^k + 6$ with initial conditions $a_0 = 20$, $a_1 = 60$.

39. Use generating functions to find an explicit formula for the Fibonacci numbers.

***40.** **a)** Show that if n is a positive integer, then

$$\binom{-1/2}{n} = \frac{\binom{2n}{n}}{(-4)^n}.$$

b) Use the extended binomial theorem and part (a) to show that the coefficient of x^n in the expansion of $(1 - 4x)^{-1/2}$ is $\binom{2n}{n}$ for all nonnegative integers n.

***41.** (Calculus required) Let $\{C_n\}$ be the sequence of Catalan numbers, that is, the solution to the recurrence relation $C_n = \sum_{k=0}^{n-1} C_k C_{n-1-k}$ with $C_0 = C_1 = 1$ (see Example 8 in Section 5.1).

a) Show that if $G(x)$ is the generating function for the sequence of Catalan numbers, then $xG(x)^2 - G(x) + 1 = 0$. Conclude (using the initial conditions) that $G(x) = (1 - \sqrt{1 - 4x})/(2x)$.

b) Use Exercise 40 to conclude that

$$G(x) = \sum_{n=0}^{\infty} \frac{1}{n+1}\binom{2n}{n}x^n,$$

so that

$$C_n = \frac{1}{n+1}\binom{2n}{n}.$$

42. Use generating functions to prove Pascal's identity: $C(n, r) = C(n - 1, r) + C(n - 1, r - 1)$ when n and r are positive integers with $r < n$. [*Hint:* Use the identity $(1 + x)^n = (1 + x)^{n-1} + x(1 + x)^{n-1}$.]

43. Use generating functions to prove Vandermonde's identity: $C(m + n, r) = \sum_{k=0}^{r} C(m, r - k)C(n, k)$, whenever m, n, and r are nonnegative integers with r not exceeding either m or n. [*Hint:* Look at the coefficient of x^r in both sides of $(1 + x)^{m+n} = (1 + x)^m(1 + x)^n$.]

44. This exercise shows how to use generating functions to derive a formula for the sum of the first n squares.

- **a)** Show that $(x^2 + x)/(1 - x)^4$ is the generating function for the sequence $\{a_n\}$, where $a_n = 1^2 + 2^2 + \cdots + n^2$.
- **b)** Use part (a) to find an explicit formula for the sum $1^2 + 2^2 + \cdots + n^2$.

The **exponential generating function** for the sequence $\{a_n\}$ is the series

$$\sum_{n=0}^{\infty} \frac{a_n}{n!}x^n.$$

For example, the exponential generating function for the sequence 1, 1, 1, … is the function $\sum_{n=0}^{\infty} x^n/n! = e^x$. (You will find this particular series useful in the following exercises.) Note that e^x is the (ordinary) generating function for the sequence 1, 1, 1/2!, 1/3!, 1/4!, … .

45. Find a closed form for the exponential generating function for the sequence $\{a_n\}$, where

- **a)** $a_n = 2$.
- **b)** $a_n = (-1)^n$.
- **c)** $a_n = 3^n$.
- **d)** $a_n = n + 1$.
- **e)** $a_n = 1/(n + 1)$.

46. Find a closed form for the exponential generating function for the sequence $\{a_n\}$, where

- **a)** $a_n = (-2)^n$.
- **b)** $a_n = -1$.
- **c)** $a_n = n$.
- **d)** $a_n = n(n - 1)$.
- **e)** $a_n = 1/((n + 1)(n + 2))$.

47. Find the sequence with each of the following functions as its exponential generating function.

- **a)** $f(x) = e^{-x}$
- **b)** $f(x) = 3x^{2x}$
- **c)** $f(x) = e^{3x} - 3e^{2x}$
- **d)** $f(x) = (1 - x) + e^{-2x}$
- **e)** $f(x) = e^{-2x} - (1/(1 - x))$
- **f)** $f(x) = e^{-3x} - (1 + x) + (1/(1 - 2x))$
- **g)** $f(x) = e^{x^2}$

48. Find the sequence with each of the following functions as its exponential generating function.

- **a)** $f(x) = e^{3x}$
- **b)** $f(x) = 2e^{-3x+1}$
- **c)** $f(x) = e^{4x} + e^{-4x}$
- **d)** $f(x) = (1 + 2x) + e^{3x}$
- **e)** $f(x) = e^x - (1/(1 + x))$
- **f)** $f(x) = xe^x$
- **g)** $f(x) = e^{x^3}$

49. A coding system encodes messages using strings of octal (base 8) digits. A codeword is considered valid if and only if it contains an even number of 7s.

a) Find a linear nonhomogeneous recurrence relation for the number of valid codewords of length n. What are the initial conditions?

b) Solve this recurrence relation using Theorem 6 in Section 5.2.

c) Solve this recurrence relation using generating functions.

***50.** A coding system encodes messages using strings of base 4 digits (that is, digits from the set $\{0, 1, 2, 3\}$). A codeword is valid if and only if it contains an even number of 0s and an even number of 1s. Let a_n equal the number of valid codewords length n. Furthermore, let b_n, c_n, and d_n equal the number of strings of base 4 digits of length n with an even number of 0s and an odd number of 1s, with an odd number of 0s and an even number of 1s, and with an odd number of 0s and an odd number of 1s, respectively.

a) Show that $d_n = 4^n - a_n - b_n - c_n$. Use this to show that $a_{n+1} = 2a_n + b_n + c_n$, $b_{n+1} = b_n - c_n + 4^n$, and $c_{n+1} = c_n - b_n + 4^n$.

b) What are a_1, b_1, c_1, and d_1?

c) Use parts (a) and (b) to find a_3, b_3, c_3, and d_3.

d) Use the recurrence relations in part (a), together with the initial conditions in part (b), to set up three equations relating the generating functions $A(x)$, $B(x)$, and $C(x)$ for the sequences $\{a_n\}$, $\{b_n\}$, and $\{c_n\}$, respectively.

e) Solve the system of equations from part (d) to get explicit formulae for $A(x)$, $B(x)$, and $C(x)$ and use these to get explicit formulae for a_n, b_n, c_n, and d_n.

Generating functions are useful in studying the number of different types of partitions of an integer n. A **partition** of a positive integer is a way to write this integer as the sum of positive integers where repetition is allowed and the order of the integers in the sum does not matter. For example, the partitions of 5 (with no restrictions) are $1 + 1 + 1 + 1 + 1$, $1 + 1 + 1 + 2$, $1 + 1 + 3$, $1 + 2 + 2$, $1 + 4$, $2 + 3$, and 5. Exercises 51–56 illustrate some of these uses.

51. Show that the coefficient $p(n)$ of x^n in the formal power series expansion of $1/((1-x)(1-x^2)(1-x^3)\cdots)$ equals the number of partitions of n.

52. Show that the coefficient $p_o(n)$ of x^n in the formal power series expansion of $1/((1-x)(1-x^3)(1-x^5)\cdots)$ equals the number of partitions of n into odd integers, that is, the number of ways to write n as the sum of odd positive integers, where the order does not matter and repetitions are allowed.

53. Show that the coefficient $p_d(n)$ of x^n in the formal power series expansion of $(1 + x)(1 + x^2)(1 + x^3)\cdots$ equals the number of partitions of n into distinct parts, that is, the number of ways to write n as the sum of positive integers, where the order does not matter but no repetitions are allowed.

54. Find $p_o(n)$, the number of partitions of n into odd parts with repetitions allowed, and $p_d(n)$, the number of partitions of n into distinct parts, for $1 \le n \le 8$, by writing each partition of each type for each integer.

55. Show that if n is a positive integer, then the number of partitions of n into distinct parts equals the number of partitions of n into odd parts with repetitions allowed; that is, $p_o(n) = p_d(n)$. [*Hint:* Show that the generating functions for $p_o(n)$ and $p_d(n)$ are equal.]

****56.** (Calculus required) Use the generating function of $p(n)$ to show that $p(n) \le e^{C\sqrt{n}}$ for some constant C. [Hardy and Ramanujan showed that $p(n) \sim e^{\pi\sqrt{2/3}\sqrt{n}}/(4\sqrt{3}n)$, which means that the ratio of $p(n)$ and the right-hand side approaches 1 as n approaches infinity.]

Suppose that X is a random variable on a sample space S such that $X(s)$ is a nonnegative integer for all $s \in S$. The **probability generating function** for X is

$$G_X(x) = \sum_{k=0}^{\infty} p(X(s) = k)x^k.$$

57. (Calculus required) Show that if G_X is the probability generating function for a random variable X such that $X(s)$ is a nonnegative integer for all $s \in S$, then

a) $G_X(1) = 1$. b) $E(X) = G'_X(1)$.

c) $V(X) = G''_X(1) + G'_X(1) - G'_X(1)^2$.

58. Let X be the random variable whose value is n if the first success occurs on the nth trial when independent Bernoulli trials are performed, each with probability of success p.

a) Find a closed formula for the probability generating function G_X.

b) Find the expected value and the variance of X using Exercise 57 and the closed form for the probability generating function found in part (a).

59. Let m be a positive integer. Let X_m be the random variable whose value is n if the mth success occurs on the $(n + m)$th trial when independent Bernoulli trials are performed, each with probability of success p.

a) Using Exercise 44 in the Supplementary Exercises of Chapter 4, show that the probability generating function G_{X_m} is given by $G(x)_{X_m} = p^m/(1 - qx)^m$, where $q = 1 - p$.

b) Find the expected value and the variance of X_m using Exercise 57 and the closed form for the probability generating function in part (a).

60. Show that if X and Y are independent random variables on a sample space S such that $X(s)$ and $Y(s)$ are nonnegative integers for all $s \in S$, then $G_{X+Y}(x) = G_X(x)G_Y(x)$.

5.5

Inclusion–Exclusion

INTRODUCTION

A discrete mathematics class contains 30 women and 50 sophomores. How many students in the class are either women or sophomores? This question cannot be answered unless more information is provided. Adding the number of women in the class and the number of sophomores probably does not give the correct answer, because women sophomores are counted twice. This observation shows that the number of students in the class that are either sophomores or women is the sum of the number of women and the number of sophomores in the class minus the number of women sophomores. A technique for solving such counting problems was introduced in Section 4.1. In this section we will generalize the ideas introduced in that section to solve a wider range of counting problems.

THE PRINCIPLE OF INCLUSION–EXCLUSION

How many elements are in the union of two finite sets? In Section 1.5 it was shown that the number of elements in the union of the two sets A and B is the sum of the numbers of elements in the sets minus the number of elements in their intersection. That is,

$$|A \cup B| = |A| + |B| - |A \cap B|.$$

As we showed in Section 4.1, the formula for the number of elements in the union of two sets is useful in counting problems. The following examples provide additional illustrations of the usefulness of this formula.

EXAMPLE 1 A discrete mathematics class contains 25 students majoring in computer science, 13 students majoring in mathematics, and 8 joint mathematics and computer science majors. How many students are in this class, if every student is majoring in mathematics, computer science, or both mathematics and computer science?

Solution: Let A be the set of students in the class majoring in computer science and B be the set of students in the class majoring in mathematics. Then $A \cap B$ is the set of students in the class that are joint mathematics and computer science majors. Since every student in the class is majoring in either computer science or mathematics (or both), it follows that the number of students in the class is $|A \cup B|$. Therefore,

$$\begin{aligned}|A \cup B| &= |A| + |B| - |A \cap B| \\ &= 25 + 13 - 8 \\ &= 30.\end{aligned}$$

Therefore, there are 30 students in the class. This computation is illustrated in Figure 1. ∎

EXAMPLE 2 How many positive integers not exceeding 1000 are divisible by 7 or 11?

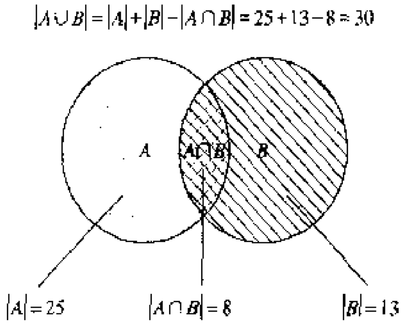


FIGURE 1 The Set of Students in a Discrete Mathematics Class.

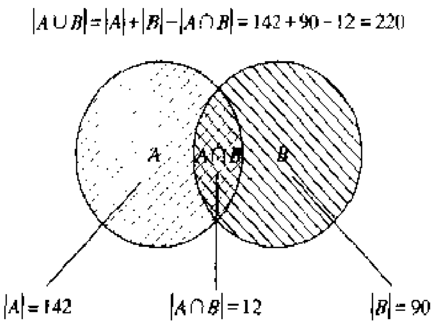


FIGURE 2 The Set of Positive Integers Not Exceeding 1000 Divisible by Either 7 or 11.

Solution: Let A be the set of positive integers not exceeding 1000 that are divisible by 7, and let B be the set of positive integers not exceeding 1000 that are divisible by 11. Then $A \cup B$ is the set of integers not exceeding 1000 that are divisible by either 7 or 11, and $A \cap B$ is the set of integers not exceeding 1000 that are divisible by both 7 and 11. From Example 2 of Section 2.3, we know that among the positive integers not exceeding 1000 there are $\lfloor 1000/7 \rfloor$ integers divisible by 7 and $\lfloor 1000/11 \rfloor$ divisible by 11. Since 7 and 11 are relatively prime, the integers divisible by both 7 and 11 are those divisible by $7 \cdot 11$. Consequently, there are $\lfloor 1000/(11 \cdot 7) \rfloor$ positive integers not exceeding 1000 that are divisible by both 7 and 11. It follows that there are

$$\begin{aligned}|A \cup B| &= |A| + |B| - |A \cap B|\\ &= \left\lfloor \frac{1000}{7} \right\rfloor + \left\lfloor \frac{1000}{11} \right\rfloor - \left\lfloor \frac{1000}{7 \cdot 11} \right\rfloor\\ &= 142 + 90 - 12\\ &= 220\end{aligned}$$

positive integers not exceeding 1000 that are divisible by either 7 or 11. This computation is illustrated in Figure 2. ■

The next example shows how to find the number of elements in a finite universal set that are outside the union of two sets.

EXAMPLE 3 Suppose that there are 1807 freshmen at your school. Of these, 453 are taking a course in computer science, 567 are taking a course in mathematics, and 299 are taking courses in both computer science and mathematics. How many are not taking a course in either computer science or in mathematics?

Solution: To find the number of freshmen who are not taking a course in either mathematics or computer science, subtract the number that are taking a course in either of these subjects from the total number of freshmen. Let A be the set of all freshmen taking a course in computer science, and let B be the set of all freshmen taking a course in

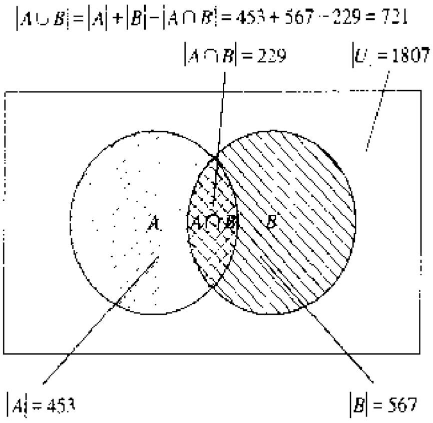


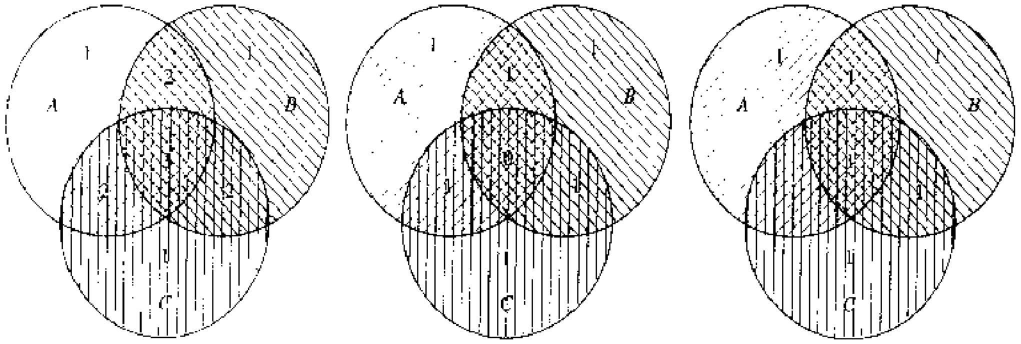
FIGURE 3 **The Set of Freshmen Not Taking a Course in Either Computer Science or Mathematics.**

mathematics. It follows that $|A| = 453$, $|B| = 567$, and $|A \cap B| = 299$. The number of freshmen taking a course in either computer science or mathematics is

$$|A \cup B| = |A| + |B| - |A \cap B| = 453 + 567 - 299 = 721.$$

Consequently, there are $1807 - 721 = 1086$ freshmen who are not taking a course in computer science or mathematics. This is illustrated in Figure 3. ■

Later in this section it will be shown how the number of elements in the union of a finite number of sets can be found. The result that will be developed is called the **principle of inclusion–exclusion.** Before considering unions of n sets, where n is any positive integer, a formula for the number of elements in the union of three sets A, B, and C will be derived. To construct this formula note that $|A| + |B| + |C|$ counts each element that is in exactly one of the three sets once, elements that are in exactly two of the sets twice, and elements in all three sets three times. This is illustrated in the first panel in Figure 4.



(a) Count of elements by $|A| + |B| + |C|$

(b) Count of elements by $|A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C|$

(c) Count of elements by $|A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$

FIGURE 4 **Finding a Formula for the Number of Elements in the Union of Three Sets.**

To remove the overcount of elements in more than one of the sets, subtract the number of elements in the intersections of all pairs of the three sets, obtaining

$$|A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C|.$$

This expression still counts elements that occur in exactly one of the sets once. An element that occurs in exactly two of the sets is also counted exactly once, since this element will occur in one of the three intersections of sets taken two at a time. However, those elements that occur in all three sets will be counted zero times by this expression, since they occur in all three intersections of sets taken two at a time. This is illustrated in the second panel in Figure 4.

To remedy this undercount, add the number of elements in the intersection of all three sets. This final expression counts each element once, whether it is in one, two, or three of the sets. Thus,

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|.$$

This formula is illustrated in the third panel of Figure 4.

The following examples illustrate how this formula can be used.

EXAMPLE 4 A total of 1232 students have taken a course in Spanish, 879 have taken a course in French, and 114 have taken a course in Russian. Further, 103 have taken courses in both Spanish and French, 23 have taken courses in both Spanish and Russian, and 14 have taken courses in both French and Russian. If 2092 students have taken at least one of Spanish, French, and Russian, how many students have taken a course in all three languages?

Solution: Let S be the set of students who have taken a course in Spanish, F the set of students who have taken a course in French, and R the set of students who have taken a course in Russian. Then

$$|S| = 1232, \qquad |F| = 879, \qquad |R| = 114,$$
$$|S \cap F| = 103, \qquad |S \cap R| = 23, \qquad |F \cap R| = 14,$$

and

$$|S \cup F \cup R| = 2092.$$

Inserting these quantities into the equation

$$|S \cup F \cup R| = |S| + |F| + |R| - |S \cap F| - |S \cap R| - |F \cap R| + |S \cap F \cap R|$$

gives

$$2092 = 1232 + 879 + 114 - 103 - 23 - 14 + |S \cap F \cap R|.$$

Solving for $|S \cap F \cap R|$ shows that $|S \cap F \cap R| = 7$. Therefore, there are seven students who have taken courses in Spanish, French, and Russian. This is illustrated in Figure 5. ∎

We will now state and prove the inclusion–exclusion principle, which tells us how many elements are in the union of a finite number of finite sets.

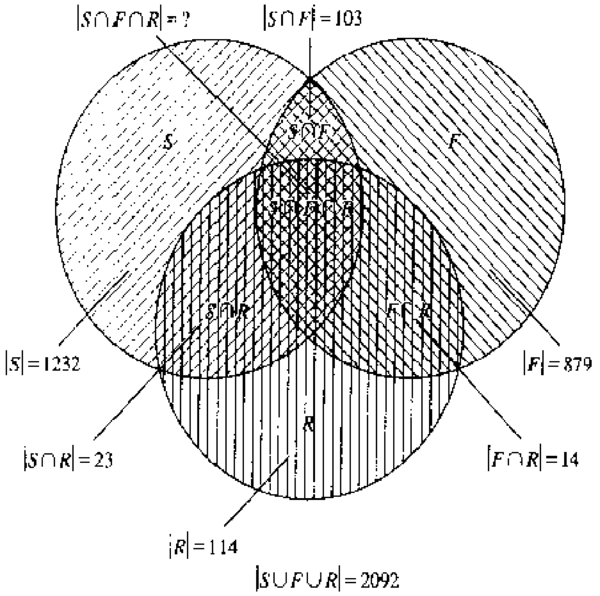


FIGURE 5 **The Set of Students Who Have Taken Courses in Spanish, French, and Russian.**

THEOREM 1 **The Principle of Inclusion–Exclusion** Let $A_1, A_2, \ldots, A_n$ be finite sets. Then

$$|A_1 \cup A_2 \cup \cdots \cup A_n| = \sum_{1 \le i \le n} |A_i| - \sum_{1 \le i < j \le n} |A_i \cap A_j|$$
$$+ \sum_{1 \le i < j < k \le n} |A_i \cap A_j \cap A_k| - \cdots + (-1)^{n+1} |A_1 \cap A_2 \cap \cdots \cap A_n|.$$

Proof: We will prove the formula by showing that an element in the union is counted exactly once by the right-hand side of the equation. Suppose that a is a member of exactly r of the sets $A_1, A_2, \ldots, A_n$ where $1 \le r \le n$. This element is counted $C(r, 1)$ times by $\Sigma|A_i|$. It is counted $C(r, 2)$ times by $\Sigma|A_i \cap A_j|$. In general, it is counted $C(r, m)$ times by the summation involving m of the sets A_i. Thus, this element is counted exactly

$$C(r, 1) - C(r, 2) + C(r, 3) - \cdots + (-1)^{r+1} C(r, r)$$

times by the expression on the right-hand side of this equation. Our goal is to evaluate this quantity. From Theorem 7 of Section 4.3, we have

$$C(r, 0) - C(r, 1) + C(r, 2) - \cdots + (-1)^r C(r, r) = 0.$$

Hence,

$$1 = C(r, 0) = C(r, 1) - C(r, 2) + \cdots + (-1)^{r+1} C(r, r).$$

Therefore, each element in the union is counted exactly once by the expression on the right-hand side of the equation. This proves the principle of inclusion–exclusion. ☐

The inclusion–exclusion principle gives a formula for the number of elements in the union of n sets for every positive integer n. There are terms in this formula for the number of elements in the intersection of every nonempty subset of the collection of the n sets. Hence, there are $2^n - 1$ terms in this formula.

EXAMPLE 5 Give a formula for the number of elements in the union of four sets.

Solution: The inclusion–exclusion principle shows that

$$\begin{aligned}|A_1 \cup A_2 \cup A_3 \cup A_4| = {} & |A_1| + |A_2| + |A_3| + |A_4| \\ & - |A_1 \cap A_2| - |A_1 \cap A_3| - |A_1 \cap A_4| - |A_2 \cap A_3| - |A_2 \cap A_4| - |A_3 \cap A_4| \\ & + |A_1 \cap A_2 \cap A_3| + |A_1 \cap A_2 \cap A_4| + |A_1 \cap A_3 \cap A_4| + |A_2 \cap A_3 \cap A_4| \\ & - |A_1 \cap A_2 \cap A_3 \cap A_4|.\end{aligned}$$

Note that this formula contains 15 different terms, one for each nonempty subset of $\{A_1, A_2, A_3, A_4\}$. ■

Exercises

1. How many elements are in $A_1 \cup A_2$ if there are 12 elements in A_1, 18 elements in A_2, and
   a) $A_1 \cap A_2 = \emptyset$?
   b) $|A_1 \cap A_2| = 1$?
   c) $|A_1 \cap A_2| = 6$?
   d) $A_1 \subseteq A_2$?
2. There are 345 students at a college who have taken a course in calculus, 212 who have taken a course in discrete mathematics, and 188 who have taken courses in both calculus and discrete mathematics. How many students have taken a course in either calculus or discrete mathematics?
3. A survey of households in the United States reveals that 96% have at least one television set, 98% have telephone service, and 95% have telephone service and at least one television set. What percentage of households in the United States have neither telephone service nor a television set?
4. A marketing report concerning personal computers states that 650,000 owners will buy a modem for their machines next year and 1,250,000 will buy at least one software package. If the report states that 1,450,000 owners will buy either a modem or at least one software package, how many will buy both a modem and at least one software package?
5. Find the number of elements in $A_1 \cup A_2 \cup A_3$ if there are 100 elements in each set if
   a) the sets are pairwise disjoint.
   b) there are 50 common elements in each pair of sets and no elements in all three sets.
   c) there are 50 common elements in each pair of sets and 25 elements in all three sets.
   d) the sets are equal.
6. Find the number of elements in $A_1 \cup A_2 \cup A_3$ if there are 100 elements in A_1, 1000 in A_2, and 10,000 in A_3 if
   a) $A_1 \subseteq A_2$ and $A_2 \subseteq A_3$.
   b) the sets are pairwise disjoint.
   c) there are two elements common to each pair of sets and one element in all three sets.
7. There are 2504 computer science students at a school. Of these, 1876 have taken a course in Pascal, 999 have taken a course in Fortran, and 345 have taken a course in C. Further, 876 have taken courses in both Pascal and Fortran, 231 have taken courses in both Fortran and C, and 290 have taken courses in both Pascal and C. If 189 of these students have taken courses in Fortran, Pascal, and C, how many of these 2504 students have not taken a course in any of these three programming languages?
8. In a survey of 270 college students, it is found that 64 like brussels sprouts, 94 like broccoli, 58 like cauliflower, 26 like both brussels sprouts and broccoli, 28 like both brussels sprouts and cauliflower, 22 like both broccoli and cauliflower, and 14 like all three vegetables. How many of the 270 students do not like any of these vegetables?
9. How many students are enrolled in a course either in calculus, discrete mathematics, data structures, or programming languages at a school if there are 507, 292, 312, and 344 students in these courses, respectively, 14 in both calculus and data structures, 213 in both calculus and programming languages, 211 in both discrete mathematics and data structures, 43 in both discrete mathematics and programming languages, and no student may take calculus and discrete mathematics, or data structures and programming languages, concurrently?
10. Find the number of positive integers not exceeding 100 that are not divisible by 5 or by 7.

11. Find the number of positive integers not exceeding 100 that are either odd or the square of an integer.

12. Find the number of positive integers not exceeding 1000 that are either the square or the cube of an integer.

13. How many bit strings of length eight do not contain six consecutive 0s?

***14.** How many permutations of the 26 letters of the English alphabet do not contain any of the strings *fish*, *rat*, or *bird*?

15. How many permutations of the 10 digits either begin with the 3 digits 987, contain the digits 45 in the fifth and sixth positions, or end with the 3 digits 123?

16. How many elements are in the union of four sets if each of the sets has 100 elements, each pair of the sets shares 50 elements, each three of the sets share 25 elements, and there are 5 elements in all four sets?

17. How many elements are in the union of four sets if the sets have 50, 60, 70, and 80 elements, respectively, each pair of the sets has 5 elements in common, each triple of the sets has 1 common element, and no element is in all four sets?

18. How many terms are there in the formula for the number of elements in the union of 10 sets given by the principle of inclusion–exclusion?

19. Write out the explicit formula given by the principle of inclusion–exclusion for the number of elements in the union of five sets.

20. How many elements are in the union of five sets if the sets contain 10,000 elements each, each pair of sets has 1000 common elements, each triple of sets has 100 common elements, every four of the sets have 10 common elements, and there is 1 element in all five sets?

21. Write out the explicit formula given by the principle of inclusion–exclusion for the number of elements in the union of six sets when it is known that no three of these sets have a common intersection.

***22.** Prove the principle of inclusion–exclusion using mathematical induction.

23. Let E_1, E_2, and E_3 be three events from a sample space S. Find a formula for the probability of $E_1 \cup E_2 \cup E_3$.

24. Find the probability that when a coin is flipped five times tails comes up exactly three times, the first and last flips come up tails, or the second and fourth flips come up heads.

25. Find the probability that when four numbers from 1 to 100, inclusive, are picked at random with no repetitions allowed, either all are odd, all are divisible by 3, or all are divisible by 5.

26. Find a formula for the probability of the union of four events in a sample space if no three of them can occur at the same time.

27. Find a formula for the probability of the union of five events in a sample space if no four of them can occur at the same time.

28. Find a formula for the probability of the union of n events in a sample space when no two of these events can occur at the same time.

29. Find a formula for the probability of the union of n events in a sample space.

5.6

Applications of Inclusion–Exclusion

INTRODUCTION

Many counting problems can be solved using the principle of inclusion–exclusion. For instance, we can use this principle to find the number of primes less than a positive integer. Many problems can be solved by counting the number of onto functions from one finite set to another. The inclusion–exclusion principle can be used to find the number of such functions. The famous hatcheck problem can be solved using the principle of inclusion–exclusion. This problem asks for the probability that no person is given the correct hat back by a hatcheck person who gives the hats back randomly.

AN ALTERNATIVE FORM OF INCLUSION–EXCLUSION

There is an alternative form of the principle of inclusion–exclusion that is useful in counting problems. In particular, this form can be used to solve problems that ask for the number of elements in a set that have none of n properties $P_1, P_2, \ldots, P_n$.

Let A_i be the subset containing the elements that have property P_i. The number of elements with all the properties $P_{i_1}, P_{i_2}, \ldots, P_{i_k}$ will be denoted by $N(P_{i_1} P_{i_2} \cdots P_{i_k})$.

Writing these quantities in terms of sets, we have

$$|A_{i_1} \cap A_{i_2} \cap \cdots \cap A_{i_k}| = N(P_{i_1} P_{i_2} \cdots P_{i_k}).$$

If the number of elements with none of the properties $P_1, P_2, \ldots, P_n$ is denoted by $N(P'_1 P'_2 \cdots P'_n)$ and the number of elements in the set is denoted by N, it follows that

$$N(P'_1 P'_2 \cdots P'_n) = N - |A_1 \cup A_2 \cup \cdots \cup A_n|.$$

From the inclusion–exclusion principle, we see that

$$N(P'_1 P'_2 \cdots P'_n) = N - \sum_{1 \le i \le n} N(P_i) + \sum_{1 \le i < j \le n} N(P_i P_j) - \sum_{1 \le i < j < k \le n} N(P_i P_j P_k)$$
$$+ \cdots + (-1)^n N(P_1 P_2 \cdots P_n).$$

The following example shows how the principle of inclusion–exclusion can be used to determine the number of solutions in integers of an equation with constraints.

EXAMPLE 1 How many solutions does

$$x_1 + x_2 + x_3 = 11$$

have, where x_1, x_2, and x_3 are nonnegative integers with $x_1 \le 3$, $x_2 \le 4$, and $x_3 \le 6$?

Solution: To apply the principle of inclusion–exclusion, let a solution have property P_1 if $x_1 > 3$, property P_2 if $x_2 > 4$, and property P_3 if $x_3 > 6$. The number of solutions satisfying the inequalities $x_1 \le 3$, $x_2 \le 4$, and $x_3 \le 6$ is

$$N(P'_1 P'_2 P'_3) = N - N(P_1) - N(P_2) - N(P_3) + N(P_1 P_2) + N(P_1 P_3) + N(P_2 P_3)$$
$$- N(P_1 P_2 P_3).$$

Using the same techniques as in Example 6 of Section 4.6, it follows that

- N = total number of solutions $= C(3 + 11 - 1, 11) = 78$,
- $N(P_1)$ = (number of solutions with $x_1 \ge 4$) $= C(3 + 7 - 1, 7) = C(9, 7) = 36$,
- $N(P_2)$ = (number of solutions with $x_2 \ge 5$) $= C(3 + 6 - 1, 6) = C(8, 6) = 28$,
- $N(P_3)$ = (number of solutions with $x_3 \ge 7$) $= C(3 + 4 - 1, 4) = C(6, 4) = 15$,
- $N(P_1 P_2)$ = (number of solutions with $x_1 \ge 4$ and $x_2 \ge 5$) $= C(3 + 2 - 1, 2) = C(4, 2) = 6$,
- $N(P_1 P_3)$ = (number of solutions with $x_1 \ge 4$ and $x_3 \ge 7$) $= C(3 + 0 - 1, 0) = 1$,
- $N(P_2 P_3)$ = (number of solutions with $x_2 \ge 5$ and $x_3 \ge 7$) $= 0$,
- $N(P_1 P_2 P_3)$ = (number of solutions with $x_1 \ge 4$, $x_2 \ge 5$, and $x_3 \ge 7$) $= 0$.

Inserting these quantities into the formula for $N(P'_1 P'_2 P'_3)$ shows that the number of solutions with $x_1 \le 3$, $x_2 \le 4$, and $x_3 \le 6$ equals

$$N(P'_1 P'_2 P'_3) = 78 - 36 - 28 - 15 + 6 + 1 + 0 - 0 = 6. \quad \blacksquare$$

THE SIEVE OF ERATOSTHENES

The principle of inclusion–exclusion can be used to find the number of primes not exceeding a specified positive integer. Recall that a composite integer is divisible by a prime not exceeding its square root. So, to find the number of primes not exceeding 100, first note that composite integers not exceeding 100 must have a prime factor not exceeding 10. Because the only primes less than 10 are 2, 3, 5, and 7, the primes not exceeding 100 are these four primes and those positive integers greater than 1 and not exceeding 100 that are divisible by none of 2, 3, 5, or 7. To apply the principle of inclusion–exclusion, let P_1 be the property that an integer is divisible by 2, let P_2 be the property that an integer is divisible by 3, let P_3 be the property that an integer is divisible by 5, and let P_4 be the property that an integer is divisible by 7. Thus, the number of primes not exceeding 100 is

$$4 + N(P_1'P_2'P_3'P_4').$$

Since there are 99 positive integers greater than 1 and not exceeding 100, the principle of inclusion–exclusion shows that

$$\begin{aligned} N(P_1'P_2'P_3'P_4') = 99 &- N(P_1) - N(P_2) - N(P_3) - N(P_4) \\ &+ N(P_1P_2) + N(P_1P_3) + N(P_1P_4) + N(P_2P_3) \\ &+ N(P_2P_4) + N(P_3P_4) \\ &- N(P_1P_2P_3) - N(P_1P_2P_4) - N(P_1P_3P_4) - N(P_2P_3P_4) \\ &+ N(P_1P_2P_3P_4). \end{aligned}$$

The number of integers not exceeding 100 (and greater than 1) that are divisible by all the primes in a subset of $\{2, 3, 5, 7\}$ is $\lfloor 100/N \rfloor$, where N is the product of the primes in this subset. (This follows since any two of these primes have no common factor.) Consequently,

$$\begin{aligned} N(P_1'P_2'P_3'P_4') = 99 &- \left\lfloor \frac{100}{2} \right\rfloor - \left\lfloor \frac{100}{3} \right\rfloor - \left\lfloor \frac{100}{5} \right\rfloor - \left\lfloor \frac{100}{7} \right\rfloor \\ &+ \left\lfloor \frac{100}{2\cdot 3} \right\rfloor + \left\lfloor \frac{100}{2\cdot 5} \right\rfloor + \left\lfloor \frac{100}{2\cdot 7} \right\rfloor + \left\lfloor \frac{100}{3\cdot 5} \right\rfloor \\ &+ \left\lfloor \frac{100}{3\cdot 7} \right\rfloor + \left\lfloor \frac{100}{5\cdot 7} \right\rfloor \\ &- \left\lfloor \frac{100}{2\cdot 3\cdot 5} \right\rfloor - \left\lfloor \frac{100}{2\cdot 3\cdot 7} \right\rfloor - \left\lfloor \frac{100}{2\cdot 5\cdot 7} \right\rfloor - \left\lfloor \frac{100}{3\cdot 5\cdot 7} \right\rfloor \\ &+ \left\lfloor \frac{100}{2\cdot 3\cdot 5\cdot 7} \right\rfloor \\ = 99 - 50 - 33 - 20 - 14 + 16 + 10 + 7 + 6 + 4 + 2& \\ - 3 - 2 - 1 - 0 + 0& \\ = 21.& \end{aligned}$$

Hence, there are $4 + 21 = 25$ primes not exceeding 100.

web

The **sieve of Eratosthenes** is used to find all primes not exceeding a specified positive integer. For instance, the following procedure is used to find the primes not

---

web

Eratosthenes (276–194 B.C.E.). It is known that Eratosthenes was born in Cyrene, a Greek colony west of Egypt, and spent time studying at Plato's Academy in Athens. We also know that King Ptolemy II invited Eratosthenes to Alexandria to tutor his son and that later Eratosthenes became chief librarian at the famous library at Alexandria, a central repository of ancient wisdom. Eratosthenes was an extremely versatile scholar, writing on mathematics, geography, astronomy, history, philosophy, and literary criticism. Besides his work in mathematics, he is most noted for his chronology of ancient history and for his famous measurement of the size of the earth.

TABLE 1 **The Sieve of Eratosthenes.**

Integers Divisible by 2 Other than 2 Receive an Underline

1	2	3	$\underline{4}$	5	$\underline{6}$	7	$\underline{8}$	9	$\underline{10}$
11	$\underline{12}$	13	$\underline{14}$	15	$\underline{16}$	17	$\underline{18}$	19	$\underline{20}$
21	$\underline{22}$	23	$\underline{24}$	25	$\underline{26}$	27	$\underline{28}$	29	$\underline{30}$
31	$\underline{32}$	33	$\underline{34}$	35	$\underline{36}$	37	$\underline{38}$	39	$\underline{40}$
41	$\underline{42}$	43	$\underline{44}$	45	$\underline{46}$	47	$\underline{48}$	49	$\underline{50}$
51	$\underline{52}$	53	$\underline{54}$	55	$\underline{56}$	57	$\underline{58}$	59	$\underline{60}$
61	$\underline{62}$	63	$\underline{64}$	65	$\underline{66}$	67	$\underline{68}$	69	$\underline{70}$
71	$\underline{72}$	73	$\underline{74}$	75	$\underline{76}$	77	$\underline{78}$	79	$\underline{80}$
81	$\underline{82}$	83	$\underline{84}$	85	$\underline{86}$	87	$\underline{88}$	89	$\underline{90}$
91	$\underline{92}$	93	$\underline{94}$	95	$\underline{96}$	97	$\underline{98}$	99	$\underline{100}$

Integers Divisible by 3 Other than 3 Receive an Underline

1	2	3	$\underline{4}$	5	$\underline{6}$	7	$\underline{8}$	$\underline{9}$	$\underline{10}$
11	$\underline{12}$	13	$\underline{14}$	$\underline{15}$	$\underline{16}$	17	$\underline{18}$	19	$\underline{20}$
$\underline{21}$	$\underline{22}$	23	$\underline{24}$	25	$\underline{26}$	$\underline{27}$	$\underline{28}$	29	$\underline{30}$
31	$\underline{32}$	$\underline{33}$	$\underline{34}$	35	$\underline{36}$	37	$\underline{38}$	$\underline{39}$	$\underline{40}$
41	$\underline{42}$	43	$\underline{44}$	$\underline{45}$	$\underline{46}$	47	$\underline{48}$	49	$\underline{50}$
$\underline{51}$	$\underline{52}$	53	$\underline{54}$	55	$\underline{56}$	$\underline{57}$	$\underline{58}$	59	$\underline{60}$
61	$\underline{62}$	$\underline{63}$	$\underline{64}$	65	$\underline{66}$	67	$\underline{68}$	$\underline{69}$	$\underline{70}$
71	$\underline{72}$	73	$\underline{74}$	$\underline{75}$	$\underline{76}$	77	$\underline{78}$	79	$\underline{80}$
$\underline{81}$	$\underline{82}$	83	$\underline{84}$	85	$\underline{86}$	$\underline{87}$	$\underline{88}$	89	$\underline{90}$
91	$\underline{92}$	$\underline{93}$	$\underline{94}$	95	$\underline{96}$	97	$\underline{98}$	$\underline{99}$	$\underline{100}$

Integers Divisible by 5 Other than 5 Receive an Underline

1	2	3	$\underline{4}$	5	$\underline{6}$	7	$\underline{8}$	$\underline{9}$	$\underline{10}$
11	$\underline{12}$	13	$\underline{14}$	$\underline{15}$	$\underline{16}$	17	$\underline{18}$	19	$\underline{20}$
$\underline{21}$	$\underline{22}$	23	$\underline{24}$	$\underline{25}$	$\underline{26}$	$\underline{27}$	$\underline{28}$	29	$\underline{30}$
31	$\underline{32}$	$\underline{33}$	$\underline{34}$	$\underline{35}$	$\underline{36}$	37	$\underline{38}$	$\underline{39}$	$\underline{40}$
41	$\underline{42}$	43	$\underline{44}$	$\underline{45}$	$\underline{46}$	47	$\underline{48}$	49	$\underline{50}$
$\underline{51}$	$\underline{52}$	53	$\underline{54}$	$\underline{55}$	$\underline{56}$	$\underline{57}$	$\underline{58}$	59	$\underline{60}$
61	$\underline{62}$	$\underline{63}$	$\underline{64}$	$\underline{65}$	$\underline{66}$	67	$\underline{68}$	$\underline{69}$	$\underline{70}$
71	$\underline{72}$	73	$\underline{74}$	$\underline{75}$	$\underline{76}$	77	$\underline{78}$	79	$\underline{80}$
$\underline{81}$	$\underline{82}$	83	$\underline{84}$	$\underline{85}$	$\underline{86}$	$\underline{87}$	$\underline{88}$	89	$\underline{90}$
91	$\underline{92}$	$\underline{93}$	$\underline{94}$	$\underline{95}$	$\underline{96}$	97	$\underline{98}$	$\underline{99}$	$\underline{100}$

Integers Divisible by 7 Other than 7 Receive an Underline; Integers in Color Are Prime

1	2	3	$\underline{4}$	5	$\underline{6}$	7	$\underline{8}$	$\underline{9}$	$\underline{10}$
11	$\underline{12}$	13	$\underline{14}$	$\underline{15}$	$\underline{16}$	17	$\underline{18}$	19	$\underline{20}$
$\underline{21}$	$\underline{22}$	23	$\underline{24}$	$\underline{25}$	$\underline{26}$	$\underline{27}$	$\underline{28}$	29	$\underline{30}$
31	$\underline{32}$	$\underline{33}$	$\underline{34}$	$\underline{35}$	$\underline{36}$	37	$\underline{38}$	$\underline{39}$	$\underline{40}$
41	$\underline{42}$	43	$\underline{44}$	$\underline{45}$	$\underline{46}$	47	$\underline{48}$	$\underline{49}$	$\underline{50}$
$\underline{51}$	$\underline{52}$	53	$\underline{54}$	$\underline{55}$	$\underline{56}$	$\underline{57}$	$\underline{58}$	59	$\underline{60}$
61	$\underline{62}$	$\underline{63}$	$\underline{64}$	$\underline{65}$	$\underline{66}$	67	$\underline{68}$	$\underline{69}$	$\underline{70}$
71	$\underline{72}$	73	$\underline{74}$	$\underline{75}$	$\underline{76}$	$\underline{77}$	$\underline{78}$	79	$\underline{80}$
$\underline{81}$	$\underline{82}$	83	$\underline{84}$	$\underline{85}$	$\underline{86}$	$\underline{87}$	$\underline{88}$	89	$\underline{90}$
$\underline{91}$	$\underline{92}$	$\underline{93}$	$\underline{94}$	$\underline{95}$	$\underline{96}$	97	$\underline{98}$	$\underline{99}$	$\underline{100}$

exceeding 100. First the integers that are divisible by 2, other than 2, are deleted. Since 3 is the first integer greater than 2 that is left, all those integers divisible by 3, other than 3, are deleted. Since 5 is the next integer left after 3, those integers divisible by 5, other than 5, are deleted. The next integer left is 7, so those integers divisible by 7, other than 7, are deleted. Since all composite integers not exceeding 100 are divisible by 2, 3, 5, or 7, all remaining integers except 1 are prime. In Table 1, the panels display those integers deleted at each stage, where each integer divisible by 2, other than 2, is underlined in the first panel, each integer divisible by 3, other than 3, is underlined in the second panel, each integer divisible by 5, other than 5, is underlined in the third panel, and each integer divisible by 7, other than 7, is underlined in the fourth panel. The integers not underlined are the primes not exceeding 100.

THE NUMBER OF ONTO FUNCTIONS

The principle of inclusion–exclusion can also be used to determine the number of onto functions from a set with m elements to a set with n elements. First consider the following example.

EXAMPLE 2 How many onto functions are there from a set with six elements to a set with three elements?

Solution: Suppose that the elements in the codomain are b_1, b_2, and b_3. Let P_1, P_2, and P_3 be the properties that b_1, b_2, and b_3 are not in the range of the function, respectively. Note that a function is onto if and only if it has none of the properties P_1, P_2, or P_3. By the inclusion–exclusion principle it follows that the number of onto functions from a set with six elements to a set with three elements is

$$N(P_1'P_2'P_3') = N - [N(P_1) + N(P_2) + N(P_3)] \\ + [N(P_1P_2) + N(P_1P_3) + N(P_2P_3)] - N(P_1P_2P_3),$$

where N is the total number of functions from a set with six elements to one with three elements. We will evaluate each of the terms on the right-hand side of this equation.

From Example 8 of Section 4.1, it follows that $N = 3^6$. Note that $N(P_i)$ is the number of functions that do not have b_i in their range. Hence, there are two choices for the value of the function at each element of the domain. Therefore, $N(P_i) = 2^6$. Furthermore, there are $C(3, 1)$ terms of this kind. Note that $N(P_iP_j)$ is the number of functions that do not have b_i and b_j in their range. Hence, there is only one choice for the value of the function at each element of the domain. Therefore, $N(P_iP_j) = 1^6 = 1$. Furthermore, there are $C(3, 2)$ terms of this kind. Also, note that $N(P_1P_2P_3) = 0$, since this term is the number of functions that have none of b_1, b_2, and b_3 in their range. Clearly, there are no such functions. Therefore, the number of onto functions from a set with six elements to one with three elements is

$$3^6 - C(3, 1)2^6 + C(3, 2)1^6 = 729 - 192 + 3 = 540.$$ ■

The general result that tells us how many onto functions there are from a set with m elements to one with n elements will now be stated. The proof of this result is left as an exercise for the reader.

THEOREM 1 **Let m and n be positive integers with $m \geq n$. Then, there are**

$$n^m - C(n, 1)(n-1)^m + C(n, 2)(n-2)^m - \cdots + (-1)^{n-1}C(n, n-1) \cdot 1^m$$

onto functions from a set with m elements to a set with n elements.

One of the many different applications of Theorem 1 will now be described.

EXAMPLE 3 How many ways are there to assign five different jobs to four different employees if every employee is assigned at least one job?

Solution: Consider the assignment of jobs as a function from the set of five jobs to the set of four employees. An assignment where every employee gets at least one job is the same as an onto function from the set of jobs to the set of employees. Hence, by

Theorem 1 it follows that there are

$$4^5 - C(4,1)3^5 + C(4,2)2^5 - C(4,3)1^5 = 1024 - 972 + 192 - 4 = 240$$

ways to assign the jobs so that each employee is assigned at least one job. ■

DERANGEMENTS

The principle of inclusion–exclusion will be used to count the permutations of n objects that leave no objects in their original positions. Consider the following example.

EXAMPLE 4 The Hatcheck Problem A new employee checks the hats of n people at a restaurant, forgetting to put claim check numbers on the hats. When customers return for their hats, the checker gives them back hats chosen at random from the remaining hats. What is the probability that no one receives the correct hat? ■

Remark: The answer is the number of ways the hats can be arranged so that there is no hat in its original position divided by $n!$, the number of permutations of n hats. We will return to this example after we find the number of permutations of n objects that leave no objects in their original position.

web

A **derangement** is a permutation of objects that leaves no object in its original position. To solve the problem posed in Example 4 we will need to determine the number of derangements of a set of n objects.

EXAMPLE 5 The permutation 21453 is a derangement of 12345 because no number is left in its original position. However, 21543 is not a derangement of 12345, because this permutation leaves 4 fixed. ■

Let D_n denote the number of derangements of n objects. For instance, $D_3 = 2$, since the derangements of 123 are 231 and 312. We will evaluate D_n, for all positive integers n, using the principle of inclusion–exclusion.

THEOREM 2 **The number of derangements of a set with n elements is**

$$D_n = n!\left[1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \cdots + (-1)^n\frac{1}{n!}\right].$$

Historical Note: In **rencontres** (matches), an old French card game, the 52 cards in a deck are laid out in a row. The cards of a second deck are laid out with one card of the second deck on top of each card of the first deck. The score is determined by counting the number of matching cards in the two decks. In 1708 Pierre Raymond de Montmort (1678–1719) posed **le problème de rencontres:** What is the probability that no matches take place in the game of rencontres? The solution to Montmort's problem is the probability that a randomly selected permutation of 52 objects is a derangement, namely, $D_{52}/52!$, which, as we will see, is approximately $1/e$.

Proof: Let a permutation have property P_i if it fixes element i. The number of derangements is the number of permutations having none of the properties P_i for $i = 1, 2, \ldots, n$, or

$$D_n = N(P_1'P_2' \cdots P_n').$$

Using the principle of inclusion–exclusion, it follows that

$$D_n = N - \sum_i N(P_i) + \sum_{i<j} N(P_iP_j) - \sum_{i<j<k} N(P_iP_jP_k) + \cdots + (-1)^n N(P_1P_2 \cdots P_n),$$

where N is the number of permutations of n elements. This equation states that the number of permutations that fix no elements equals the total number of permutations, less the number that fix at least one element, plus the number that fix at least two elements, less the number that fix at least three elements, and so on. All the quantities that occur on the right-hand side of this equation will now be found.

First, note that $N = n!$, since N is simply the total number of permutations of n elements. Also, $N(P_i) = (n-1)!$. This follows from the product rule, since $N(P_i)$ is the number of permutations that fix element i, so that the ith position of the permutation is determined, but each of the remaining positions can be filled arbitrarily. Similarly,

$$N(P_iP_j) = (n-2)!,$$

since this is the number of permutations that fix elements i and j, but where the other $n - 2$ elements can be arranged arbitrarily. In general, note that

$$N(P_{i_1}P_{i_2} \cdots P_{i_m}) = (n-m)!,$$

because this is the number of permutations that fix elements $i_1, i_2, \ldots, i_m$, but where the other $n - m$ elements can be arranged arbitrarily. Because there are $C(n, m)$ ways to choose m elements from n, it follows that

$$\sum_{1 \le i \le n} N(P_i) = C(n, 1)(n-1)!,$$

$$\sum_{1 \le i < j \le n} N(P_iP_j) = C(n, 2)(n-2)!,$$

and in general,

$$\sum_{1 \le i_1 < i_2 < \cdots < i_m \le n} N(P_{i_1}P_{i_2} \cdots P_{i_m}) = C(n, m)(n-m)!.$$

Consequently, inserting these quantities into our formula for D_n gives

$$\begin{aligned} D_n &= n! - C(n, 1)(n-1)! + C(n, 2)(n-2)! - \cdots + (-1)^n C(n, n)(n-n)! \\ &= n! - \frac{n!}{1!(n-1)!}(n-1)! + \frac{n!}{2!(n-2)!}(n-2)! - \cdots + (-1)^n \frac{n!}{n!\,0!}0!. \end{aligned}$$

Simplifying this expression gives

$$D_n = n!\left[1 - \frac{1}{1!} + \frac{1}{2!} - \cdots + (-1)^n \frac{1}{n!}\right].$$

□

TABLE 2 The Probability of a Derangement.

n	2	3	4	5	6	7
$D_n/n!$	0.50000	0.33333	0.37500	0.36667	0.36806	0.36786

It is now simple to find D_n for a given positive integer n. For instance, using Theorem 2, it follows that

$$D_3 = 3!\left[1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!}\right] = 6\left(1 - 1 + \frac{1}{2} - \frac{1}{6}\right) = 2,$$

as we have previously remarked.

The solution of the problem in Example 4 can now be given.

Solution: The probability that no one receives the correct hat is $D_n/n!$. By Theorem 2, this probability is

$$\frac{D_n}{n!} = 1 - \frac{1}{1!} + \frac{1}{2!} - \cdots + (-1)^n\frac{1}{n!}.$$

The values of this probability for $2 \leq n \leq 7$ are displayed in Table 2.

Using methods from calculus it can be shown that

$$e^{-1} = 1 - \frac{1}{1!} + \frac{1}{2!} - \cdots + (-1)^n\frac{1}{n!} + \cdots \approx 0.368.$$

Since this is an alternating series with terms tending to zero, it follows that as n grows without bound, the probability that no one receives the correct hat converges to $e^{-1} \approx 0.368$. In fact, this probability can be shown to be within $1/(n+1)!$ of e^{-1}. ■

Exercises

1. Suppose that in a bushel of 100 apples there are 20 that have worms in them and 15 that have bruises. Only those apples with neither worms nor bruises can be sold. If there are 10 bruised apples that have worms in them, how many of the 100 apples can be sold?
2. Of 1000 applicants for a mountain-climbing trip in the Himalayas, 450 get altitude sickness, 622 are not in good enough shape, and 30 have allergies. An applicant qualifies if and only if this applicant does not get altitude sickness, is in good shape, and does not have allergies. If there are 111 applicants who get altitude sickness and are not in good enough shape, 14 who get altitude sickness and have allergies, 18 who are not in good enough shape and have allergies, and 9 who get altitude sickness, are not in good enough shape, and have allergies, how many applicants qualify?
3. How many solutions does the equation $x_1 + x_2 + x_3 = 13$ have where x_1, x_2, and x_3 are nonnegative integers less than 6?
4. Find the number of solutions of the equation $x_1 + x_2 + x_3 + x_4 = 17$ where x_i, $i = 1, 2, 3, 4$, are nonnegative integers such that $x_1 \leq 3$, $x_2 \leq 4$, $x_3 \leq 5$, and $x_4 \leq 8$.
5. Find the number of primes less than 200 using the principle of inclusion–exclusion.
6. An integer is called **squarefree** if it is not divisible by the square of a positive integer greater than 1. Find the number of squarefree positive integers less than 100.
7. How many positive integers less than 10,000 are not the second or higher power of an integer?
8. How many onto functions are there from a set with seven elements to one with five elements?
9. How many ways are there to distribute six different toys to three different children such that each child gets at least one toy?
10. In how many ways can eight distinct balls be distributed into three distinct urns if each urn must contain at least one ball?
11. In how many ways can seven different jobs be assigned to four different employees so that each employee is assigned at least one job and the most difficult job is assigned to the best employee?
12. List all the derangements of $\{1, 2, 3, 4\}$.
13. How many derangements are there of a set with seven elements?

14. What is the probability that none of 10 people receives the correct hat if a hatcheck person hands their hats back randomly?

15. A machine that inserts letters into envelopes goes haywire and inserts letters randomly into envelopes. What is the probability that in a group of 100 letters

a) no letter is put into the correct envelope?
b) exactly 1 letter is put into the correct envelope?
c) exactly 98 letters are put into the correct envelope?
d) exactly 99 letters are put into the correct envelope?
e) all letters are put into the correct envelope?

16. A group of n students is assigned seats for each of two classes in the same classroom. How many ways can these seats be assigned if no student is assigned the same seat for both classes?

***17.** How many ways can the digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 be arranged so that no even digit is in its original position?

***18.** Use a combinatorial argument to show that the sequence $\{D_n\}$, where D_n denotes the number of derangements of n objects, satisfies the recurrence relation

$$D_n = (n-1)(D_{n-1} + D_{n-2})$$

for $n \geq 2$.

***19.** Use Exercise 18 to show that

$$D_n = nD_{n-1} + (-1)^n$$

for $n \geq 1$.

20. Use Exercise 19 to find an explicit formula for D_n.

21. For which positive integers n is D_n, the number of derangements of n objects, even?

22. Suppose that p and q are distinct primes. Use the principle of inclusion–exclusion to find $\phi(pq)$, the number of integers not exceeding pq that are relatively prime to pq.

***23.** Use the principle of inclusion–exclusion to derive a formula for $\phi(n)$ when the prime factorization of n is

$$n = p_1^{a_1} p_2^{a_2} \cdots p_m^{a_m}.$$

***24.** Show that if n is a positive integer, then

$$n! = C(n,0)D_n + C(n,1)D_{n-1} + \cdots + C(n,n-1)D_1 + C(n,n)D_0,$$

where D_k is the number of derangements of k objects.

25. How many derangements of $\{1, 2, 3, 4, 5, 6\}$ begin with the integers 1, 2, and 3, in some order?

26. How many derangements of $\{1, 2, 3, 4, 5, 6\}$ end with the integers 1, 2, and 3, in some order?

27. Prove Theorem 1.

Key Terms and Results

TERMS

recurrence relation: a formula expressing terms of a sequence, except for some initial terms, as a function of one or more previous terms of the sequence

initial conditions for a recurrence relation: the values of the terms of a sequence satisfying the recurrence relation before this relation takes effect

linear homogeneous recurrence relation with constant coefficients: a recurrence relation that expresses the terms of a sequence, except initial terms, as a linear combination of previous terms

characteristic roots of a linear homogeneous recurrence relation with constant coefficients: the roots of the polynomial associated with a linear homogeneous recurrence relation with constant coefficients

linear nonhomogeneous recurrence relation with constant coefficients: a recurrence relation that expresses the terms of a sequence, except for initial terms, as a linear combination of previous terms plus a function which is not identically zero that depends only on the index

divide-and-conquer algorithm: an algorithm that solves a problem recursively by splitting it into a fixed number of smaller problems of the same type

generating function of a sequence: the formal series that has the nth term of the sequence as the coefficient of x^n

sieve of Eratosthenes: a procedure for finding the primes less than a specified positive integer

derangement: a permutation of objects such that no object is in its original place

RESULTS

The formula for the number of elements in the union of two finite sets:

$$|A \cup B| = |A| + |B| - |A \cap B|$$

The formula for the number of elements in the union of three finite sets:

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

The principle of inclusion–exclusion:

$$|A_1 \cup A_2 \cup \cdots \cup A_n| = \sum_{1 \leq i \leq n} |A_i| - \sum_{1 \leq i < j \leq n} |A_i \cap A_j| + \sum_{1 \leq i < j < k \leq n} |A_i \cap A_j \cap A_k| - \cdots + (-1)^{n+1}|A_1 \cap A_2 \cap \cdots \cap A_n|$$

The number of onto functions from a set with m elements to a set with n elements:

$$n^m - C(n,1)(n-1)^m + C(n,2)(n-2)^m - \cdots + (-1)^{n-1}C(n,n-1)\cdot 1^m$$

The number of derangements of n objects:

$$D_n = n!\left[1 - \frac{1}{1!} + \frac{1}{2!} - \cdots + (-1)^n\frac{1}{n!}\right]$$

Review Questions

1. a) What is a recurrence relation?

b) Find a recurrence relation for the amount of money that will be in an account after n years if $1,000,000 is deposited in an account yielding 9%.

2. Explain how the Fibonacci numbers are used to solve Fibonacci's problem about rabbits.

3. a) Find a recurrence relation for the number of steps needed to solve the Tower of Hanoi puzzle.

b) Show how this recurrence relation can be solved using iteration.

4. a) Explain how to find a recurrence relation for the number of bit strings of length n not containing two consecutive 1s.

b) Describe another counting problem that has a solution satisfying the same recurrence relation.

5. Define a linear homogeneous recurrence relation of degree k.

6. a) Explain how to solve linear homogeneous recurrence relations of degree two.

b) Solve the recurrence relation $a_n = 13a_{n-1} - 22a_{n-2}$ for $n \geq 2$ if $a_0 = 3$ and $a_1 = 15$.

c) Solve the recurrence relation $a_n = 14a_{n-1} - 49a_{n-2}$ for $n \geq 2$ if $a_0 = 3$ and $a_1 = 35$.

7. a) Explain how to find $f(b^k)$ where k is a positive integer if $f(n)$ satisfies the divide-and-conquer recurrence relation $f(n) = af(n/b) + g(n)$ whenever b divides the positive integer n.

b) Find $f(256)$ if $f(n) = 3f(n/4) + 5n/4$ and $f(1) = 7$.

8. a) Derive a divide-and-conquer recurrence relation for the number of comparisons used to find a number in a list using a binary search.

b) Give a big-O estimate for the number of comparisons used by a binary search from the divide-and-conquer recurrence relation you gave in (a) using Theorem 1 in Section 5.3.

9. a) Give a formula for the number of elements in the union of three sets.

b) Explain why this formula is valid.

c) Explain how to use the formula from (a) to find the number of integers not exceeding 1000 that are divisible by 6, 10, or 15.

d) Explain how to use the formula from (a) to find the number of solutions in nonnegative integers to the equation $x_1 + x_2 + x_3 + x_4 = 22$ with $x_1 < 8$, $x_2 < 6$, and $x_3 < 5$.

10. a) Give a formula for the number of elements in the union of four sets and explain why it is valid.

b) Suppose the sets A_1, A_2, A_3, and A_4 each contain 25 elements, the intersection of any two of these sets contains 5 elements, the intersection of any three of these sets contains 2 elements, and 1 element is in all four of the sets. How many elements are in the union of the four sets?

11. a) State the principle of inclusion–exclusion.

b) Outline a proof of this principle.

12. Explain how the principle of inclusion–exclusion can be used to count the number of onto functions from a set with m elements to a set with n elements.

13. a) How can you count the number of ways to assign m jobs to n employees so that each employee is assigned at least one job?

b) How many ways are there to assign seven jobs to three employees so that each employee is assigned at least one job?

14. Explain how the inclusion–exclusion principle can be used to count the number of primes not exceeding the positive integer n.

15. a) Define a derangement.

b) Why is counting the number of ways a hatcheck person can return hats to n people, so that no one receives the correct hat, the same as counting the number of derangements of n objects?

c) Explain how to count the number of derangements of n objects.

Supplementary Exercises

1. A group of 10 people begin a chain letter, with each person sending the letter to 4 other people. Each of these people sends the letter to 4 additional people.

a) Find a recurrence relation for the number of letters sent at the nth stage of this chain letter, if no person ever receives more than one letter.

 b) What are the initial conditions for the recurrence relation in part (a)?
 c) How many letters are sent at the nth stage of the chain letter?

2. A nuclear reactor has created 18 grams of a particular radioactive isotope. Every hour 1% of a radioactive isotope decays.
 a) Set up a recurrence relation for the amount of this isotope left after n hours.
 b) What are the initial conditions for the recurrence relation in part (a)?
 c) Solve this recurrence relation.

3. Every hour the U.S. government prints 10,000 more \$1 bills, 4000 more \$5 bills, 3000 more \$10 bills, 2500 more \$20 bills, 1000 more \$50 bills, and the same number of \$100 bills as it did the previous hour. In the initial hour 1000 of each bill were produced.
 a) Set up a recurrence relation for the amount of money produced in the nth hour.
 b) What are the initial conditions for the recurrence relation in part (a)?
 c) Solve the recurrence relation for the amount of money produced in the nth hour.
 d) Set up a recurrence relation for the total amount of money produced in the first n hours.
 e) Solve the recurrence relation for the total amount of money produced in the first n hours.

4. Suppose that every hour there are two new bacteria in a colony for each bacterium that was present the previous hour, and that all bacteria 2 hours old die. The colony starts with 100 new bacteria.
 a) Set up a recurrence relation for the number of bacteria present after n hours.
 b) What is the solution of this recurrence relation?
 c) When will the colony contain more than 1 million bacteria?

5. Messages are sent over a communications channel using two different signals. One signal requires 2 microseconds for transmittal, and the other signal requires 3 microseconds for transmittal. Each signal of a message is followed immediately by the next signal.
 a) Find a recurrence relation for the number of different signals that can be sent in n microseconds.
 b) What are the initial conditions of the recurrence relation in part (a)?
 c) How many different messages can be sent in 12 microseconds?

6. A small post office has only 4-cent stamps, 6-cent stamps, and 10-cent stamps. Find a recurrence relation for the number of ways to form postage of n cents with these stamps if the order that the stamps are used matters. What are the initial conditions for this recurrence relation?

7. How many ways are there to form the following postages using the rules described in Exercise 6?
 a) 12 cents
 b) 14 cents
 c) 18 cents
 d) 22 cents

8. Find the solutions of the simultaneous system of congruences

$$a_n = a_{n-1} + b_{n-1}$$
$$b_n = a_{n-1} - b_{n-1}$$

with $a_0 = 1$ and $b_0 = 2$.

9. Solve the recurrence relation $a_n = a_{n-1}^2/a_{n-2}$ if $a_0 = 1$ and $a_1 = 2$. (*Hint:* Take logarithms of both sides to obtain a recurrence relation for the sequence $\log a_n$, $n = 0, 1, 2, \ldots$.)

***10.** Solve the recurrence relation $a_n = a_{n-1}^3 a_{n-2}^2$ if $a_0 = 2$ and $a_1 = 2$. (See the hint for Exercise 9.)

11. Find the solution of the recurrence relation $a_n = 3a_{n-1} - 3a_{n-2} + a_{n-3} + 1$ if $a_0 = 2$, $a_1 = 4$, and $a_2 = 8$.

12. Find the solution of the recurrence relation $a_n = 3a_{n-1} - 3a_{n-2} + a_{n-3}$ if $a_0 = 2$, $a_1 = 2$, and $a_2 = 4$.

***13.** Suppose that in Example 4 of Section 5.1 a pair of rabbits leaves the island after reproducing twice. Find a recurrence relation for the number of rabbits on the island in the middle of the nth month.

14. Find the solution to the recurrence relation $f(n) = 3f(n/5) + 2n^4$, when n is divisible by 5, for $n = 5^k$, where k is a positive integer and $f(1) = 1$.

15. Estimate the size of f in Exercise 14 if f is an increasing function.

16. Find a recurrence relation that describes the number of comparisons used by the following algorithm: Find the largest and second largest elements of a sequence of n numbers recursively by splitting the sequence into two subsequences with an equal number of terms, or where there is one more term in one subsequence than in the other, at each stage. Stop when subsequences with two terms are reached.

17. Estimate the number of comparisons used by the algorithm described in Exercise 16.

Let $\{a_n\}$ be a sequence of real numbers. The **forward differences** of this sequence are defined recursively as follows: The **first forward difference** is $\Delta a_n = a_{n+1} - a_n$; the $(k+1)$**th forward difference** $\Delta^{k+1}a_n$ is obtained from $\Delta^k a_n$ by $\Delta^{k+1}a_n = \Delta^k a_{n+1} - \Delta^k a_n$.

18. Find Δa_n where
 a) $a_n = 3$.
 b) $a_n = 4n + 7$.
 c) $a_n = n^2 + n + 1$.

19. Let $a_n = 3n^3 + n + 2$. Find $\Delta^k a_n$ where k equals
 a) 2. **b)** 3. **c)** 4.

***20.** Suppose that $a_n = P(n)$ where P is a polynomial of degree d. Prove that $\Delta^{d+1}a_n = 0$ for all nonnegative integers n.

21. Let $\{a_n\}$ and $\{b_n\}$ be sequences of real numbers. Show that

$$\Delta(a_n b_n) = a_{n+1}(\Delta b_n) + b_n(\Delta a_n).$$

22. Show that if $F(x)$ and $G(x)$ are the generating functions for the sequences $\{a_k\}$ and $\{b_k\}$, respectively, and c and d are real numbers, then $(cF + dG)(x)$ is the generating function for $\{ca_k + db_k\}$.
23. (Calculus required) This exercise shows how generating functions can be used to solve the recurrence relation $(n + 1)a_{n+1} = a_n + (1/n!)$ for $n \geq 0$ with initial condition $a_0 = 1$.
    a) Let $G(x)$ be the generating function for $\{a_n\}$. Show that $G'(x) = G(x) + e^x$ and $G(0) = 1$.
    b) Show from part (a) that $(e^{-x}G(x))' = 1$, and conclude that $G(x) = xe^x + e^x$.
    c) Use part (b) to find a closed form for a_n.
24. Suppose that 14 students get an A on the first exam in a discrete mathematics class, and 18 get an A on the second exam. If 22 students received an A on either the first exam or the second exam, how many students received an A on both exams?
25. There are 323 farms in Monmouth County that have at least one of horses, cows, and sheep. If 224 have horses, 85 have cows, 57 have sheep, and 18 farms have all three types of animals, how many farms have exactly two of these three types of animals?
26. Queries to a database of student records at a college produced the following data: There are 2175 students at the college, 1675 of these are not freshmen, 1074 students have taken a course in calculus, 444 students have taken a course in discrete mathematics, 607 students are not freshmen and have taken calculus, 350 students have taken calculus and discrete mathematics, 201 students are not freshmen and have taken discrete mathematics, and 143 students are not freshmen and have taken both calculus and discrete mathematics. Can all the responses to the queries be correct?
27. Students in the school of mathematics at a university major in one or more of the following four areas: applied mathematics (AM), pure mathematics (PM), operations research (OR), and computer science (CS). How many students are in this school if (including joint majors) there are 23 students majoring in AM; 17 students majoring in PM; 44 in OR; 63 in CS; 5 in AM and PM; 8 in AM and CS; 4 in AM and OR; 6 in PM and CS; 5 in PM and OR; 14 in OR and CS; 2 in PM, OR, and CS; 2 in AM, OR, and CS; 1 in PM, AM, and OR; 1 in PM, AM, and CS; and 1 in all four fields.
28. How many terms are needed when the inclusion–exclusion principle is used to express the number of elements in the union of seven sets if no more than five of these sets have a common element?
29. How many solutions in positive integers are there to the equation $x_1 + x_2 + x_3 = 20$ with $2 < x_1 < 6$, $6 < x_2 < 10$, and $0 < x_3 < 5$?
30. How many positive integers less than 1,000,000 are
    a) divisible by 2, 3, or 5?
    b) not divisible by 7, 11, or 13?
    c) divisible by 3 but not by 7?
31. How many positive integers less than 200 are
    a) second or higher powers of integers?
    b) either second or higher powers of integers or primes?
    c) not divisible by the square of an integer greater than 1?
    d) not divisible by the cube of an integer greater than 1?
    e) not divisible by three or more primes?

*32. How many ways are there to assign six different jobs to three different employees if the hardest job is assigned to the most experienced employee and the easiest job is assigned to the least experienced employee?

33. What is the probability that exactly one person is given back the correct hat by a hatcheck person who gives n people their hats back at random?
34. How many bit strings of length six do not contain four consecutive 1s?
35. What is the probability that a bit string of length six contains at least four 1s?

Computer Projects

WRITE PROGRAMS WITH THE FOLLOWING INPUT AND OUTPUT.

1. Given a positive integer n, list all the moves required in the Tower of Hanoi puzzle to move n disks from one peg to another according to the rules of the puzzle.
2. Given a positive integer n and an integer k with $1 \leq k \leq n$, list all the moves used by the Frame–Stewart algorithm (described in the preamble to Exercise 48 of Section 5.1) to move n disks from one peg to another using four pegs according to the rules of the puzzle.
3. Given a positive integer n, list all the bit sequences of length n that do not have a pair of consecutive 0s.
4. Given a positive integer n, write out all ways to parenthesize the product of $n + 1$ variables.
5. Given a recurrence relation $a_n = c_1a_{n-1} + c_2a_{n-2}$ where c_1 and c_2 are real numbers, initial conditions $a_0 = C_0$ and $a_1 = C_1$, and a positive integer k, find a_k using iteration.

6. Given a recurrence relation $a_n = c_1a_{n-1} + c_2a_{n-2}$ and initial conditions $a_0 = C_0$ and $a_1 = C_1$, determine the unique solution.
7. Given a recurrence relation of the form $f(n) = af(n/b) + c$, where a is a real number, b is a positive integer, and c is a real number, and a positive integer k, find $f(b^k)$ using iteration.
8. Given the number of elements in the intersection of three sets, the number of elements in each pairwise intersection of these sets, and the number of elements in each set, find the number of elements in their union.
9. Given a positive integer n, produce the formula for the number of elements in the union of n sets.
10. Given positive integers m and n, find the number of onto functions from a set with m elements to a set with n elements.
11. Given a positive integer n, list all the derangements of the set $\{1, 2, 3, \ldots, n\}$.

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. Find the exact value of f_{100}, f_{500}, and f_{1000} where f_n is the nth Fibonacci number.
2. Find the smallest Fibonacci number greater than 1,000,000, greater than 1,000,000,000, and greater than 1,000,000,000,000.
3. Find as many prime Fibonacci numbers as you can. It is unknown whether there are infinitely many of these.
4. Write out all the moves required to solve the Tower of Hanoi puzzle with 10 disks.
5. Write out all the moves required to use the Frame–Stewart algorithm to move 20 disks from one peg to another peg using four pegs according to the rules of the Reve's puzzle.
6. Verify the Frame conjecture for solving the Reve's puzzle for n disks for as many integers n as possible by showing that the puzzle cannot be solved using fewer moves than are made by the Frame–Stewart algorithm with the optimal choice of k.
7. Compute the number of operations required to multiply two integers with n bits for various integers n including 16, 64, 256, and 1024 using the fast multiplication described in Section 5.3 and the standard algorithm for multiplying integers (Algorithm 4 in Section 2.4).
8. Compute the number of operations required to multiply two $n \times n$ matrices for various integers n including 4, 16, 64, and 128 using the fast matrix multiplication described in Section 5.3 and the standard algorithm for multiplying matrices (Algorithm 1 in Section 2.6).
9. Use the sieve of Eratosthenes to find all the primes not exceeding 1000.
10. Find the number of primes not exceeding 10,000 using the method described in Section 5.6 to find the number of primes not exceeding 100.
11. List all the derangements of $\{1, 2, 3, 4, 5, 6, 7, 8\}$.
12. Compute the probability that a permutation of n objects is a derangement for all positive integers not exceeding 20 and determine how quickly these probabilities approach the number e.

Writing Projects

RESPOND TO THE FOLLOWING WITH ESSAYS USING OUTSIDE SOURCES.

1. Find the original source where Fibonacci presented his puzzle about modeling rabbit populations. Discuss this problem and other problems posed by Fibonacci and give some information about Fibonacci himself.
2. Explain how the Fibonacci numbers arise in a variety of applications, such as in phyllotaxis, the study of arrangement of leaves in plants, in the study of reflections by mirrors, and so on.
3. Describe different variations of the Tower of Hanoi puzzle, including those with more than three pegs (including the Reve's puzzle discussed in the text and exercises), those where disk moves are restricted, and those where disks may have the same size. Include what is known about the number of moves required to solve each variation.
4. Discuss as many different problems as possible where the Catalan numbers arise.
5. Look up the definition of the *lucky numbers*. Explain how they are found using a sieve technique similar to the sieve of Eratosthenes. Find all the lucky numbers less than 1000.
6. Describe how sieve methods are used in number theory. What kind of results have been established using such methods?

7. Look up the rules of the old French card game of *rencontres*. Describe these rules and describe the work of Pierre Raymond de Montmort on *le problème de rencontres*.
8. Describe how exponential generating functions can be used to solve a variety of counting problems.
9. Describe the Polyá theory of counting and the kind of counting problems that can be solved using this theory.
10. The *problème des ménages* (the problem of the households) asks for the number of ways to arrange n couples around a table so that the sexes alternate and no husband and wife are seated together. Explain the method used by E. Lucas to solve this problem.
11. Explain how *rook polynomials* can be used to solve counting problems.

6 Relations

Relationships between elements of sets occur in many contexts. Every day we deal with relationships such as those between a business and its telephone number, an employee and his or her salary, a person and a relative, and so on. In mathematics we study relationships such as those between a positive integer and one that it divides, an integer and one that it is congruent to modulo 5, a real number and one that is larger than it, and so on. Relationships such as that between a program and a variable it uses and that between a computer language and a valid statement in this language often arise in computer science.

Relationships between elements of sets are represented using the structure called a relation. Relations can be used to solve problems such as determining which pairs of cities are linked by airline flights in a network, finding a viable order for the different phases of a complicated project, or producing a useful way to store information in computer databases.

6.1

Relations and Their Properties

INTRODUCTION

The most direct way to express a relationship between elements of two sets is to use ordered pairs made up of two related elements. For this reason, sets of ordered pairs are called binary relations. In this section we introduce the basic terminology used to describe binary relations. Later in this chapter we will use relations to solve problems involving communications networks, project scheduling, and identifying elements in sets with common properties.

DEFINITION 1. Let A and B be sets. A *binary relation from A to B* is a subset of $A \times B$.

In other words, a binary relation from A to B is a set R of ordered pairs where the first element of each ordered pair comes from A and the second element comes from B. We use the notation $a\,R\,b$ to denote that $(a, b) \in R$ and $a \not R\, b$ to denote that $(a, b) \notin R$. Moreover, when (a, b) belongs to R, a is said to be **related to** b by R.

Binary relations represent relationships between the elements of two sets. We will introduce n-ary relations, which express relationships among elements of more than two sets, later in this chapter. We will omit the word *binary* when there is no danger of confusion.

The following are examples of relations.

EXAMPLE 1 Let A be the set of students in your school, and let B be the set of courses. Let R be the relation that consists of those pairs (a, b) where a is a student enrolled in course b. For instance, if Jason Goodfriend and Deborah Sherman are enrolled in CS518, which is Discrete Mathematics, the pairs (Jason Goodfriend, CS518) and (Deborah Sherman, CS518) belong to R. If Jason Goodfriend is also enrolled in CS510, which is Data Structures, then the pair (Jason Goodfriend, CS510) is also in R. However, if Deborah Sherman is not enrolled in CS510, then the pair (Deborah Sherman, CS510) is not in R. ■

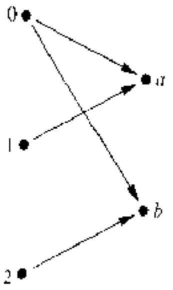
EXAMPLE 2 Let A be the set of all cities, and let B be the set of the 50 states in the United States of America. Define the relation R by specifying that (a, b) belongs to R if city a is in state b. For instance, (Boulder, Colorado), (Bangor, Maine), (Ann Arbor, Michigan), (Cupertino, California), and (Red Bank, New Jersey) are in R. ■

EXAMPLE 3 Let $A = \{0, 1, 2\}$ and $B = \{a, b\}$. Then $\{(0, a), (0, b), (1, a), (2, b)\}$ is a relation from A to B. This means, for instance, that $0 \, R \, a$, but that $1 \not{R} \, b$. Relations can be represented graphically, as shown in Figure 1, using arrows to represent ordered pairs. Another way to represent this relation is to use a table, which is also done in Figure 1. We will discuss representations of relations in more detail in Section 6.3. ■

FUNCTIONS AS RELATIONS

Recall that a function f from a set A to a set B (as defined in Section 1.6) assigns a unique element of B to each element of A. The graph of f is the set of ordered pairs (a, b) such that $b = f(a)$. Since the graph of f is a subset of $A \times B$, it is a relation from A to B. Moreover, the graph of a function has the property that every element of A is the first element of exactly one ordered pair of the graph.

Conversely, if R is a relation from A to B such that every element in A is the first element of exactly one ordered pair of R, then a function can be defined with R as its graph. This can be done by assigning to an element a of A the unique element $b \in B$ such that $(a, b) \in R$.



R	a	b
0	×	×
1	×	
2		×

FIGURE 1 **Displaying the Ordered Pairs in the Relation R from Example 3.**

A relation can be used to express a one-to-many relationship between the elements of the sets A and B, where an element of A may be related to more than one element of B. A function represents a relation where exactly one element of B is related to each element of A.

RELATIONS ON A SET

Relations from a set A to itself are of special interest.

DEFINITION 2. A *relation on* the set A is a relation from A to A.

In other words, a relation on a set A is a subset of $A \times A$.

EXAMPLE 4 Let A be the set $\{1, 2, 3, 4\}$. Which ordered pairs are in the relation $R = \{(a, b) \mid a \text{ divides } b\}$?

Solution: Since (a, b) is in R if and only if a and b are positive integers not exceeding 4 such that a divides b, we see that

$$R = \{(1, 1), (1, 2), (1, 3), (1, 4), (2, 2), (2, 4), (3, 3), (4, 4)\}.$$

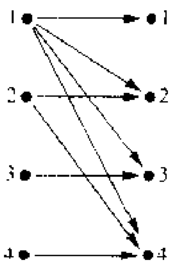
The pairs in this relation are displayed both graphically and in tabular form in Figure 2. ■

Next, some examples of relations on the set of integers will be given.

EXAMPLE 5 Consider the following relations on the set of integers:

$$R_1 = \{(a, b) \mid a \leq b\}$$
$$R_2 = \{(a, b) \mid a > b\}$$
$$R_3 = \{(a, b) \mid a = b \text{ or } a = -b\}$$
$$R_4 = \{(a, b) \mid a = b\}$$
$$R_5 = \{(a, b) \mid a = b + 1\}$$
$$R_6 = \{(a, b) \mid a + b \leq 3\}.$$

Which of these relations contain each of the pairs $(1, 1)$, $(1, 2)$, $(2, 1)$, $(1, -1)$, and $(2, 2)$?



R	1	2	3	4
1	×	×	×	×
2		×		×
3			×	
4				×

FIGURE 2 **Displaying the Ordered Pairs in the Relation R from Example 4.**

Remark: Unlike the relations in Examples 1–4, these are relations on an infinite set.

Solution: The pair (1, 1) is in R_1, R_3, R_4, and R_6; (1, 2) is in R_1 and R_6; (2, 1) is in R_2, R_5, and R_6; (1, −1) is in R_2, R_3, and R_6; and finally, (2, 2) is in R_1, R_3, and R_4. ■

It is not hard to determine the number of relations on a finite set, since a relation on a set A is simply a subset of $A \times A$.

EXAMPLE 6 How many relations are there on a set with n elements?

Solution: A relation on a set A is a subset of $A \times A$. Since $A \times A$ has n^2 elements when A has n elements, and a set with m elements has 2^m subsets, there are 2^{n^2} subsets of $A \times A$. Thus, there are 2^{n^2} relations on a set with n elements. ■

PROPERTIES OF RELATIONS

There are several properties that are used to classify relations on a set. We will introduce the most important of these here.

In some relations an element is always related to itself. For instance, let R be the relation on the set of all people consisting of pairs (x, y) where x and y have the same mother and the same father. Then $x\,R\,x$ for every person x.

DEFINITION 3. A relation R on a set A is called *reflexive* if $(a, a) \in R$ for every element $a \in A$.

We see that a relation on A is reflexive if every element of A is related to itself. The following examples illustrate the concept of a reflexive relation.

EXAMPLE 7 Consider the following relations on {1, 2, 3, 4}:

$R_1 = \{(1, 1), (1, 2), (2, 1), (2, 2), (3, 4), (4, 1), (4, 4)\}$
$R_2 = \{(1, 1), (1, 2), (2, 1)\}$
$R_3 = \{(1, 1), (1, 2), (1, 4), (2, 1), (2, 2), (3, 3), (4, 1), (4, 4)\}$
$R_4 = \{(2, 1), (3, 1), (3, 2), (4, 1), (4, 2), (4, 3)\}$
$R_5 = \{(1, 1), (1, 2), (1, 3), (1, 4), (2, 2), (2, 3), (2, 4), (3, 3), (3, 4), (4, 4)\}$
$R_6 = \{(3, 4)\}$.

Which of these relations are reflexive?

Solution: The relations R_3 and R_5 are reflexive since they both contain all pairs of the form (a, a), namely, (1, 1), (2, 2), (3, 3), and (4, 4). The other relations are not reflexive since they do not contain all of these ordered pairs. In particular, R_1, R_2, R_4, and R_6 are not reflexive since (3, 3) is not in any of these relations. ■

EXAMPLE 8 Which of the relations from Example 5 are reflexive?

Solution: The reflexive relations from this example are R_1 (since $a \leq a$ for every integer a), R_3, and R_4. For each of the other relations in this example it is easy to find a pair of the form (a, a) that is not in the relation. (This is left as an exercise for the reader.) ■

EXAMPLE 9 Is the "divides" relation on the set of positive integers reflexive?

Solution: Since $a \mid a$ whenever a is a positive integer, the "divides" relation is reflexive. ■

In some relations an element is related to a second element if and only if the second element is also related to the first element. The relation consisting of pairs (x, y) where x and y are students at your school with at least one common class has this property. Other relations have the property that if an element is related to a second element, then this second element is not related to the first. The relation consisting of the pairs (x, y) where x and y are students at your school where x has a higher grade point average than y has this property.

DEFINITION 4. A relation R on a set A is called *symmetric* if $(b, a) \in R$ whenever $(a, b) \in R$, for $a, b \in A$. A relation R on a set A such that $(a, b) \in R$ and $(b, a) \in R$ only if $a = b$, for $a, b \in A$, is called *antisymmetric*.

That is, a relation is symmetric if and only if a is related to b implies that b is related to a. A relation is antisymmetric if and only if there are no pairs of distinct elements a and b with a related to b and b related to a. The terms *symmetric* and *antisymmetric* are not opposites, since a relation can have both of these properties or may lack both of them (see Exercise 6 at the end of this section). A relation cannot be both symmetric and antisymmetric if it contains some pair of the form (a, b) where $a \neq b$.

EXAMPLE 10 Which of the relations from Example 7 are symmetric and which are antisymmetric?

Solution: The relations R_2 and R_3 are symmetric, because in each case (b, a) belongs to the relation whenever (a, b) does. For R_2, the only thing to check is that both $(2, 1)$ and $(1, 2)$ are in the relation. For R_3, it is necessary to check that both $(1, 2)$ and $(2, 1)$ belong to the relation, and $(1, 4)$ and $(4, 1)$ belong to the relation. The reader should verify that none of the other relations is symmetric. This is done by finding a pair (a, b) so that it is in the relation but (b, a) is not.

R_4, R_5, and R_6 are all antisymmetric. For each of these relations there is no pair of elements a and b with $a \neq b$ such that both (a, b) and (b, a) belong to the relation. The reader should verify that none of the other relations is antisymmetric. This is done by finding a pair (a, b) with $a \neq b$ so that (a, b) and (b, a) are both in the relation. ■

EXAMPLE 11 Which of the relations from Example 5 are symmetric and which are antisymmetric?

Solution: The relations R_3, R_4, and R_6 are symmetric. R_3 is symmetric, for if $a = b$ or $a = -b$, then $b = a$ or $b = -a$. R_4 is symmetric since $a = b$ implies that $b = a$.

R_6 is symmetric since $a + b \leq 3$ implies that $b + a \leq 3$. The reader should verify that none of the other relations is symmetric.

The relations R_1, R_2, R_4, and R_5 are antisymmetric. R_1 is antisymmetric because the inequalities $a \leq b$ and $b \leq a$ imply that $a = b$. R_2 is antisymmetric since it is impossible for $a > b$ and $b > a$. R_4 is antisymmetric, since two elements are related with respect to R_4 if and only if they are equal. R_5 is antisymmetric since it is impossible that $a = b + 1$ and $b = a + 1$. The reader should verify that none of the other relations is antisymmetric. ■

EXAMPLE 12 Is the "divides" relation on the set of positive integers symmetric? Is it antisymmetric?

Solution: This relation is not symmetric since $1 \mid 2$, but $2 \nmid 1$. It is antisymmetric, for if a and b are positive integers with $a \mid b$ and $b \mid a$, then $a = b$ (the verification of this is left as an exercise for the reader). ■

Let R be the relation consisting of all pairs (x, y) of students at your school where x has taken more credits than y. Suppose that x is related to y and y is related to z. This means that x has taken more credits than y and y has taken more credits than z. We can conclude that x has taken more credits than z, so that x is related to z. What we have shown is that R has the transitive property, which is defined as follows.

DEFINITION 5. A relation R on a set A is called *transitive* if whenever $(a, b) \in R$ and $(b, c) \in R$, then $(a, c) \in R$, for $a, b, c \in A$.

EXAMPLE 13 Which of the relations in Example 7 are transitive?

Solution: R_4, R_5, and R_6 are transitive. For each of these relations, we can show that it is transitive by verifying that if (a, b) and (b, c) belong to this relation, then (a, c) also does. For instance, R_4 is transitive, since $(3, 2)$ and $(2, 1)$, $(4, 2)$ and $(2, 1)$, $(4, 3)$ and $(3, 1)$, and $(4, 3)$ and $(3, 2)$ are the only such sets of pairs, and $(3, 1)$, $(4, 1)$, and $(4, 2)$ belong to R_4. The reader should verify that R_5 and R_6 are transitive.

R_1 is not transitive since $(3, 4)$ and $(4, 1)$ belong to R_1, but $(3, 1)$ does not. R_2 is not transitive since $(2, 1)$ and $(1, 2)$ belong to R_2, but $(2, 2)$ does not. R_3 is not transitive since $(4, 1)$ and $(1, 2)$ belong to R_3, but $(4, 2)$ does not. ■

EXAMPLE 14 Which of the relations in Example 5 are transitive?

Solution: The relations R_1, R_2, R_3, and R_4 are transitive. R_1 is transitive since $a \leq b$ and $b \leq c$ imply that $a \leq c$. R_2 is transitive since $a > b$ and $b > c$ imply that $a > c$. R_3 is transitive since $a = \pm b$ and $b = \pm c$ imply that $a = \pm c$. R_4 is clearly transitive, as the reader should verify. R_5 is not transitive since $(2, 1)$ and $(1, 0)$ belong to R_5, but $(2, 0)$ does not. R_6 is not transitive since $(2, 1)$ and $(1, 2)$ belong to R_6, but $(2, 2)$ does not. ■

EXAMPLE 15 Is the "divides" relation on the set of positive integers transitive?

Solution: Suppose that a divides b and b divides c. Then there are positive integers k and l such that $b = ak$ and $c = bl$. Hence, $c = akl$, so that a divides c. It follows that this relation is transitive. ■

The next example shows how to count the number of relations with a specified property.

EXAMPLE 16 How many *reflexive* relations are there on a set with n elements?

Solution: A relation R on a set A is a subset of $A \times A$. Consequently, a relation is determined by specifying whether each of the n^2 ordered pairs in $A \times A$ is in R. However, if R is reflexive, each of the n ordered pairs (a, a) for $a \in A$ must be in R. Each of the other $n(n - 1)$ ordered pairs of the form (a, b) where $a \neq b$ may or may not be in R. Hence, by the product rule for counting, there are $2^{n(n-1)}$ reflexive relations [this is the number of ways to choose whether each element (a, b) with $a \neq b$ belongs to R]. ■

The number of symmetric relations and the number of antisymmetric relations on a set with n elements can be found using reasoning similar to that in Example 16 (see Exercise 25 at the end of this section). Counting the transitive relations on a set with n elements is a problem beyond the scope of this book.

COMBINING RELATIONS

Since relations from A to B are subsets of $A \times B$, two relations from A to B can be combined in any way two sets can be combined. Consider the following examples.

EXAMPLE 17 Let $A = \{1, 2, 3\}$ and $B = \{1, 2, 3, 4\}$. The relations $R_1 = \{(1, 1), (2, 2), (3, 3)\}$ and $R_2 = \{(1, 1), (1, 2), (1, 3), (1, 4)\}$ can be combined to obtain

$$\begin{aligned} R_1 \cup R_2 &= \{(1, 1), (1, 2), (1, 3), (1, 4), (2, 2), (3, 3)\}, \\ R_1 \cap R_2 &= \{(1, 1)\}, \\ R_1 - R_2 &= \{(2, 2), (3, 3)\}, \\ R_2 - R_1 &= \{(1, 2), (1, 3), (1, 4)\}. \end{aligned}$$ ■

EXAMPLE 18 Let A and B be the set of all students and the set of all courses at a school, respectively. Suppose that R_1 consists of all ordered pairs (a, b), where a is a student who has taken course b, and R_2 consists of all ordered pairs (a, b), where a is a student who requires course b to graduate. What are the relations $R_1 \cup R_2$, $R_1 \cap R_2$, $R_1 \oplus R_2$, $R_1 - R_2$, and $R_2 - R_1$?

Solution: The relation $R_1 \cup R_2$ consists of all ordered pairs (a, b), where a is a student who either has taken course b or needs course b to graduate, and $R_1 \cap R_2$ is the set of all ordered pairs (a, b), where a is a student who has taken course b and needs this course to graduate. Also, $R_1 \oplus R_2$ consists of all ordered pairs (a, b), where student a has taken course b but does not need it to graduate or needs course b to graduate but has not taken it. $R_1 - R_2$ is the set of ordered pairs (a, b), where a has taken course b but does not need it to graduate; that is, b is an elective course that a has taken. $R_2 - R_1$ is the set of all ordered pairs (a, b), where b is a course that a needs to graduate but has not taken. ■

There is another way that relations are combined which is analogous to the composition of functions.

DEFINITION 6. Let R be a relation from a set A to a set B and S a relation from B to a set C. The composite of R and S is the relation consisting of ordered pairs (a, c), where $a \in A$, $c \in C$, and for which there exists an element $b \in B$ such that $(a, b) \in R$ and $(b, c) \in S$. We denote the composite of R and S by $S \circ R$.

The following example illustrates how composites of relations are formed.

EXAMPLE 19 What is the composite of the relations R and S where R is the relation from $\{1, 2, 3\}$ to $\{1, 2, 3, 4\}$ with $R = \{(1, 1), (1, 4), (2, 3), (3, 1), (3, 4)\}$ and S is the relation from $\{1, 2, 3, 4\}$ to $\{0, 1, 2\}$ with $S = \{(1, 0), (2, 0), (3, 1), (3, 2), (4, 1)\}$?

Solution: $S \circ R$ is constructed using all ordered pairs in R and ordered pairs in S, where the second element of the ordered pair in R agrees with the first element of the ordered pair in S. For example, the ordered pairs $(2, 3)$ in R and $(3, 1)$ in S produce the ordered pair $(2, 1)$ in $S \circ R$. Computing all the ordered pairs in the composite, we find

$$S \circ R = \{(1, 0), (1, 1), (2, 1), (2, 2), (3, 0), (3, 1)\}.$$ ■

The powers of a relation R can be inductively defined from the definition of a composite of two relations.

DEFINITION 7. Let R be a relation on the set A. The powers R^n, $n = 1, 2, 3, \ldots$, are defined inductively by

$$R^1 = R \quad \text{and} \quad R^{n+1} = R^n \circ R.$$

The definition shows that $R^2 = R \circ R$, $R^3 = R^2 \circ R = (R \circ R) \circ R$, and so on.

EXAMPLE 20 Let $R = \{(1, 1), (2, 1), (3, 2), (4, 3)\}$. Find the powers R^n, $n = 2, 3, 4, \ldots$.

Solution: Since $R^2 = R \circ R$, we find that $R^2 = \{(1, 1), (2, 1), (3, 1), (4, 2)\}$. Furthermore, since $R^3 = R^2 \circ R$, $R^3 = \{(1, 1), (2, 1), (3, 1), (4, 1)\}$. Additional computation shows that R^4 is the same as R^3, so $R^4 = \{(1, 1), (2, 1), (3, 1), (4, 1)\}$. It also follows that $R^n = R^3$ for $n = 5, 6, 7, \ldots$. The reader should verify this. ■

The following theorem shows that the powers of a transitive relation are subsets of this relation. It will be used in Section 6.4.

THEOREM 1 **The relation R on a set A is transitive if and only if $R^n \subseteq R$ for $n = 1, 2, 3, \ldots$.**

Proof: We first prove the if part of the theorem. Suppose that $R^n \subseteq R$ for $n = 1, 2, 3, \ldots$. In particular, $R^2 \subseteq R$. To see that this implies R is transitive, note that if $(a, b) \in R$ and $(b, c) \in R$, then by the definition of composition, $(a, c) \in R^2$. Since $R^2 \subseteq R$, this means that $(a, c) \in R$. Hence R is transitive.

We will use mathematical induction to prove the only if part of the theorem. Note that this part of the theorem is trivially true for $n = 1$.

Assume that $R^n \subseteq R$ where n is a positive integer. This is the inductive hypothesis. To complete the inductive step we must show that this implies that R^{n+1} is also a subset of R. To show this, assume that $(a, b) \in R^{n+1}$. Then, since $R^{n+1} = R^n \circ R$, there is an element x with $x \in A$ such that $(a, x) \in R$ and $(x, b) \in R^n$. The inductive hypothesis, namely, that $R^n \subseteq R$, implies that $(x, b) \in R$. Furthermore, since R is transitive, and $(a, x) \in R$ and $(x, b) \in R$, it follows that $(a, b) \in R$. This shows that $R^{n+1} \subseteq R$, completing the proof. □

Exercises

1. List the ordered pairs in the relation R from $A = \{0, 1, 2, 3, 4\}$ to $B = \{0, 1, 2, 3\}$ where $(a, b) \in R$ if and only if
  - **a)** $a = b$.
  - **b)** $a + b = 4$.
  - **c)** $a > b$.
  - **d)** $a \mid b$.
  - **e)** $\gcd(a, b) = 1$.
  - **f)** $\operatorname{lcm}(a, b) = 2$.

2. **a)** List all the ordered pairs in the relation $R = \{(a, b) \mid a \text{ divides } b\}$ on the set $\{1, 2, 3, 4, 5, 6\}$.
  - **b)** Display this relation graphically, as was done in Example 4.
  - **c)** Display this relation in tabular form, as was done in Example 4.

3. For each of the following relations on the set $\{1, 2, 3, 4\}$, decide whether it is reflexive, whether it is symmetric, whether it is antisymmetric, and whether it is transitive.
  - **a)** $\{(2, 2), (2, 3), (2, 4), (3, 2), (3, 3), (3, 4)\}$
  - **b)** $\{(1, 1), (1, 2), (2, 1), (2, 2), (3, 3), (4, 4)\}$
  - **c)** $\{(2, 4), (4, 2)\}$
  - **d)** $\{(1, 2), (2, 3), (3, 4)\}$
  - **e)** $\{(1, 1), (2, 2), (3, 3), (4, 4)\}$
  - **f)** $\{(1, 3), (1, 4), (2, 3), (2, 4), (3, 1), (3, 4)\}$

4. Determine whether the relation R on the set of all people is reflexive, symmetric, antisymmetric, and/or transitive, where $(a, b) \in R$ if and only if
  - **a)** a is taller than b.
  - **b)** a and b were born on the same day.
  - **c)** a has the same first name as b.
  - **d)** a and b have a common grandparent.

5. Determine whether the relation R on the set of all integers is reflexive, symmetric, antisymmetric, and/or transitive, where $(x, y) \in R$ if and only if
  - **a)** $x \neq y$.
  - **b)** $xy \geq 1$.
  - **c)** $x = y + 1$ or $x = y - 1$.
  - **d)** $x \equiv y \pmod 7$.
  - **e)** x is a multiple of y.
  - **f)** x and y are both negative or both nonnegative.
  - **g)** $x = y^2$.
  - **h)** $x \geq y^2$.

6. Give an example of a relation on a set that is
  - **a)** symmetric and antisymmetric.
  - **b)** neither symmetric nor antisymmetric.

☞ A relation R on the set A is **irreflexive** if for every $a \in A$, $(a, a) \notin R$. That is, R is irreflexive if no element in A is related to itself.

7. Which relations in Exercise 3 are irreflexive?

8. Which relations in Exercise 4 are irreflexive?

9. Can a relation on a set be neither reflexive nor irreflexive?

A relation R is called **asymmetric** if $(a, b) \in R$ implies that $(b, a) \notin R$.

10. Which relations in Exercise 3 are asymmetric?

11. Which relations in Exercise 4 are asymmetric?

12. Must an asymmetric relation also be antisymmetric? Must an antisymmetric relation be asymmetric? Give reasons for your answers.

13. How many different relations are there from a set with m elements to a set with n elements?

Let R be a relation from a set A to a set B. The **inverse relation** from B to A, denoted by R^{-1}, is the set of ordered pairs $\{(b, a) \mid (a, b) \in R\}$. The **complementary relation** $\overline{R}$ is the set of ordered pairs $\{(a, b) \mid (a, b) \notin R\}$.

14. Let R be the relation $R = \{(a, b) \mid a < b\}$ on the set of integers. Find
- **a)** R^{-1}.
- **b)** $\overline{R}$.

15. Let R be the relation $R = \{(a, b) \mid a \text{ divides } b\}$ on the set of positive integers. Find
- **a)** R^{-1}.
- **b)** $\overline{R}$.

16. Let R be the relation on the set of all states in the United States consisting of pairs (a, b) where state a borders state b. Find
- **a)** R^{-1}.
- **b)** $\overline{R}$.

17. Suppose that the function f from A to B is a one-to-one correspondence. Let R be the relation that equals the graph of f. That is, $R = \{(a, f(a)) \mid a \in A\}$. What is the inverse relation R^{-1}?

18. Let $R_1 = \{(1, 2), (2, 3), (3, 4)\}$ and $R_2 = \{(1, 1), (1, 2), (2, 1), (2, 2), (2, 3), (3, 1), (3, 2), (3, 3), (3, 4)\}$ be relations from $\{1, 2, 3\}$ to $\{1, 2, 3, 4\}$. Find
- **a)** $R_1 \cup R_2$.
- **b)** $R_1 \cap R_2$.
- **c)** $R_1 - R_2$.
- **d)** $R_2 - R_1$.

19. Let A be the set of students at your school and B the set of books in the school library. Let R_1 and R_2 be the relations consisting of all ordered pairs (a, b) where student a is required to read book b in a course, and where student a has read book b, respectively. Describe the ordered pairs in each of the following relations.
- **a)** $R_1 \cup R_2$
- **b)** $R_1 \cap R_2$
- **c)** $R_1 \oplus R_2$
- **d)** $R_1 - R_2$
- **e)** $R_2 - R_1$

20. Let R be the relation $\{(1, 2), (1, 3), (2, 3), (2, 4), (3, 1)\}$, and let S be the relation $\{(2, 1), (3, 1), (3, 2), (4, 2)\}$. Find $S \circ R$.

21. Let R be the relation on the set of people consisting of pairs (a, b) where a is a parent of b. Let S be the relation on the set of people consisting of pairs (a, b) where a and b are siblings (brothers or sisters). What are $S \circ R$ and $R \circ S$?

22. List the 16 different relations on the set $\{0, 1\}$.

23. How many of the 16 different relations on $\{0, 1\}$ contain the pair $(0, 1)$?

24. Which of the 16 relations on $\{0, 1\}$, which you listed in Exercise 22, are
- **a)** reflexive?
- **b)** irreflexive?
- **c)** symmetric?
- **d)** antisymmetric?
- **e)** asymmetric?
- **f)** transitive?

***25.** How many relations are there on a set with n elements that are
- **a)** symmetric?
- **b)** antisymmetric?
- **c)** asymmetric?
- **d)** irreflexive?
- **e)** reflexive and symmetric?
- **f)** neither reflexive nor irreflexive?

***26.** How many transitive relations are there on a set with n elements if

a) $n = 1$? **b)** $n = 2$? **c)** $n = 3$?

27. Find the error in the "proof" of the following "theorem."

"Theorem": Let R be a relation on a set A that is symmetric and transitive. Then R is reflexive.

"Proof": Let $a \in A$. Take an element $b \in A$ such that $(a, b) \in R$. Since R is symmetric, we also have $(b, a) \in R$. Now using the transitive property, we can conclude that $(a, a) \in R$ since $(a, b) \in R$ and $(b, a) \in R$.

28. Suppose that R and S are reflexive relations on a set A. Prove or disprove each of the following statements.
- **a)** $R \cup S$ is reflexive.
- **b)** $R \cap S$ is reflexive.
- **c)** $R \oplus S$ is irreflexive.
- **d)** $R - S$ is irreflexive.
- **e)** $S \circ R$ is reflexive.

29. Show that the relation R on a set A is symmetric if and only if $R = R^{-1}$ where R^{-1} is the inverse relation.

30. Show that the relation R on a set A is antisymmetric if and only if $R \cap R^{-1}$ is a subset of the diagonal relation $\Delta = \{(a, a) \mid a \in A\}$.

31. Show that the relation R on a set A is reflexive if and only if the inverse relation R^{-1} is reflexive.

32. Show that the relation R on a set A is reflexive if and only if the complementary relation $\overline{R}$ is irreflexive.

33. Let R be a relation that is reflexive and transitive. Prove that $R^n = R$ for all positive integers n.

34. Let R be the relation on the set $\{1, 2, 3, 4, 5\}$ containing the ordered pairs $(1, 1), (1, 2), (1, 3), (2, 3), (2, 4), (3, 1), (3, 4), (3, 5), (4, 2), (4, 5), (5, 1), (5, 2)$, and $(5, 4)$. Find

a) R^2. **b)** R^3. **c)** R^4. **d)** R^5.

35. Let R be a reflexive relation on a set A. Show that R^n is reflexive for all positive integers n.

***36.** Let R be a symmetric relation. Show that R^n is symmetric for all positive integers n.

37. Suppose that the relation R is irreflexive. Is R^2 necessarily irreflexive? Give a reason for your answer.

6.2

n-ary Relations and Their Applications

INTRODUCTION

Relationships among elements of more than two sets often arise. For instance, there is a relationship involving the name of a student, the student's major, and the student's grade point average. Similarly, there is a relationship involving the airline, flight number, starting point, destination, departure time, and arrival time of a flight. An example of such a relationship in mathematics involves three integers where the first integer is larger than the second integer, which is larger than the third. Another example is the betweenness relationship involving points on a line, such that three points are related when the second point is between the first and the third.

We will study relationships among elements from more than two sets in this section. These relationships are called ***n*-ary relations.** These relations are used to represent computer databases. These representations help us answer queries about the information stored in databases, such as: Which flights land at O'Hare Airport between 3 A.M. and 4 A.M.? Which students at your school are sophomores majoring in mathematics or computer science and have greater than a 3.0 average? Which employees of a company have worked for the company less than 5 years and make more than $50,000?

n-ARY RELATIONS

We begin with a definition.

DEFINITION 1. Let $A_1, A_2, \ldots, A_n$ be sets. An *n-ary relation* on these sets is a subset of $A_1 \times A_2 \times \cdots \times A_n$. The sets $A_1, A_2, \ldots, A_n$ are called the *domains* of the relation, and n is called its *degree*.

EXAMPLE 1 Let R be the relation consisting of triples (a, b, c) where a, b, and c are integers with $a < b < c$. Then $(1, 2, 3) \in R$, but $(2, 4, 3) \notin R$. The degree of this relation is 3. Its domains are all equal to the set of integers. ■

EXAMPLE 2 Let R be the relation consisting of 5-tuples (A, N, S, D, T) representing airplane flights, where A is the airline, N is the flight number, S is the starting point, D is the destination, and T is the departure time. For instance, if Nadir Express Airlines has flight 963 from Newark to Bangor at 15:00, then (Nadir, 963, Newark, Bangor, 15:00) belongs to R. The degree of this relation is 5, and its domains are the set of all airlines, the set of flight numbers, the set of cities, the set of cities (again), and the set of times. ■

DATABASES AND RELATIONS

The time required to manipulate information in a database depends on how this information is stored. The operations of adding and deleting records, updating records, searching for records, and combining records from overlapping databases are performed millions of times each day in a large database. Because of the importance of these operations, various methods for representing databases have been developed. We will discuss one of these methods, called the **relational data model,** based on the concept of a relation.

A database consists of **records,** which are n-tuples, made up of **fields.** The fields are the entries of the n-tuples. For instance, a database of student records may be made up of fields containing the name, student number, major, and grade point average of the student. The relational data model represents a database of records as an n-ary relation. Thus, student records are represented as 4-tuples of the form (*STUDENT NAME, ID NUMBER, MAJOR, GPA*). A sample database of six such records is:

(Ackermann, 231455, Computer Science, 3.88)

(Adams, 888323, Physics, 3.45)

(Chou, 102147, Computer Science, 3.79)

(Goodfriend, 453876, Mathematics, 3.45)

(Rao, 678543, Mathematics, 3.90)

(Stevens, 786576, Psychology, 2.99).

Relations used to represent databases are also called **tables,** since these relations are often displayed as tables. For instance, the same database of students is displayed in Table 1.

A domain of an n-ary relation is called a **primary key** when the value of the n-tuple from this domain determines the n-tuple. That is, a domain is a primary key when no two n-tuples in the relation have the same value from this domain.

Records are often added to or deleted from databases. Because of this, the property that a domain is a primary key is time-dependent. Consequently, a primary key should be chosen that remains one whenever the database is changed. This can be done by using a primary key of the **intension** of the database, which contains all the n-tuples that can ever be included in an n-ary relation representing this database.

EXAMPLE 3 Which domains are primary keys for the n-ary relation displayed in Table 1, assuming that no n-tuples will be added in the future?

TABLE 1

Student Name	*ID Number*	*Major*	*GPA*
Ackerman	231455	Computer Science	3.88
Adams	888323	Physics	3.45
Chou	102147	Computer Science	3.79
Goodfriend	453876	Mathematics	3.45
Rao	678543	Mathematics	3.90
Stevens	786576	Psychology	2.99

Solution: Since there is only one 4-tuple in this table for each student name, the domain of student names is a primary key. Similarly, the ID numbers in this table are unique, so that the domain of ID numbers is also a primary key. However, the domain of major fields of study is not a primary key, since more than one 4-tuple contains the same major field of study. The domain of grade point averages is also not a primary key, since there are two 4-tuples containing the same GPA (which ones?). ■

Combinations of domains can also uniquely identify n-tuples in an n-ary relation. When the values of a set of domains determines an n-tuple in a relation, the Cartesian product of these domains is called a **composite key.**

EXAMPLE 4 Is the Cartesian product of the domain of major fields of study and the domain of GPAs a composite key for the n-ary relation from Table 1, assuming that no n-tuples are ever added?

Solution: Since no two 4-tuples from this table have both the same major and the same GPA, this Cartesian product is a composite key. ■

Since primary and composite keys are used to identify records uniquely in a database, it is important that keys remain valid when new records are added to the database. Hence, checks should be made to ensure that every new record has values that are different in the appropriate field, or fields, from all other records in this table. For instance, it makes sense to use the student identification number as a key for student records if no two students ever have the same student identification number. A university should not use the name field as a key, since two students may have the same name (such as John Smith).

There are a variety of operations on n-ary relations that can be used to form new n-ary relations. Two of these operations will be discussed here, namely, the projection and join operations. Projections are used to form new n-ary relations by deleting the same fields in every record of the relation.

DEFINITION 2. The *projection* $P_{i_1 i_2,\ldots,i_m}$ maps the n-tuple $(a_1, a_2, \ldots, a_n)$ to the m-tuple $(a_{i_1}, a_{i_2}, \ldots, a_{i_m})$, where $m \leq n$.

In other words, the projection $P_{i_1,i_2,\ldots,i_m}$ deletes $n - m$ of the components of an n-tuple, leaving the i_1th, i_2th, …, and i_mth components.

EXAMPLE 5 What results when the projection $P_{1,3}$ is applied to the 4-tuples (2, 3, 0, 4), (Jane Doe, 234111001, Geography, 3.14), and (a_1, a_2, a_3, a_4)?

Solution: The projection $P_{1,3}$ sends these 4-tuples to (2, 0), (Jane Doe, Geography), and (a_1, a_3), respectively. ■

The following example illustrates how new relations are produced using projections.

EXAMPLE 6 What relation results when the projection $P_{1,4}$ is applied to the relation in Table 1?

Solution: When the projection $P_{1,4}$ is used, the second and third columns of the table are deleted, and pairs representing student names and grade point averages are obtained. Table 2 displays the results of this projection. ■

Fewer rows may result when a projection is applied to the table for a relation. This happens when some of the n-tuples in the relation have identical values in each of the m components of the projection, and only disagree in components deleted by the projection. For instance, consider the following example.

EXAMPLE 7 What is the table obtained when the projection $P_{1,2}$ is applied to the relation in Table 3?

Solution: Table 4 displays the relation obtained when $P_{1,2}$ is applied to Table 3. Note that there are fewer rows after this projection is applied. ■

The **join** operation is used to combine two tables into one when these tables share some identical fields. For instance, a table containing fields for airline, flight number, and gate, and another table containing fields for flight number, gate, and departure time can be combined into a table containing fields for airline, flight number, gate, and departure time.

DEFINITION 3. Let R be a relation of degree m and S a relation of degree n. The *join* $J_p(R, S)$, where $p \leq m$ and $p \leq n$, is a relation of degree $m + n - p$ that consists of all $(m + n - p)$-tuples $(a_1, a_2, \ldots, a_{m-p}, c_1, c_2, \ldots, c_p, b_1, b_2, \ldots, b_{n-p})$, where the m-tuple $(a_1, a_2, \ldots, a_{m-p}, c_1, c_2, \ldots, c_p)$ belongs to R and the n-tuple $(c_1, c_2, \ldots, c_p, b_1, b_2, \ldots, b_{n-p})$ belongs to S.

TABLE 2

Student Name	*GPA*
Ackerman	3.88
Adams	3.45
Chou	3.79
Goodfriend	3.45
Rao	3.90
Stevens	2.99

TABLE 3

Student	*Major*	*Course*
Glauser	Biology	BI 290
Glauser	Biology	MS 475
Glauser	Biology	PY 410
Marcus	Mathematics	MS 511
Marcus	Mathematics	MS 603
Marcus	Mathematics	CS 322
Miller	Computer Science	MS 575
Miller	Computer Science	CS 455

TABLE 4

Student	*Major*
Glauser	Biology
Marcus	Mathematics
Miller	Computer Science

TABLE 5

Professor	*Department*	*Course Number*
Cruz	Zoology	335
Cruz	Zoology	412
Farber	Psychology	501
Farber	Psychology	617
Grammer	Physics	544
Grammer	Physics	551
Rosen	Computer Science	518
Rosen	Mathematics	575

TABLE 6

Department	*Course Number*	*Room*	*Time*
Computer Science	518	N521	2:00 P.M.
Mathematics	575	N502	3:00 P.M.
Mathematics	611	N521	4:00 P.M.
Physics	544	B505	4:00 P.M.
Psychology	501	A100	3:00 P.M.
Psychology	617	A110	11:00 A.M.
Zoology	335	A100	9:00 A.M.
Zoology	412	A100	8:00 A.M.

In other words, the join operator J_p produces a new relation from two relations by combining all m-tuples of the first relation with all n-tuples of the second relation, where the last p components of the m-tuples agree with the first p components of the n-tuples.

EXAMPLE 8 What relation results when the join operator J_2 is used to combine the relation displayed in Tables 5 and 6?

Solution: The join J_2 produces the relation shown in Table 7. ■

There are other operators besides projections and joins that produce new relations from existing relations. A description of these operations may be found in books on database theory.

TABLE 7

Professor	*Department*	*Course Number*	*Room*	*Time*
Cruz	Zoology	335	A100	9:00 A.M.
Cruz	Zoology	412	A100	8:00 A.M.
Farber	Psychology	501	A100	3:00 P.M.
Farber	Psychology	617	A110	11:00 A.M.
Grammer	Physics	544	B505	4:00 P.M.
Rosen	Computer Science	518	N521	2:00 P.M.
Rosen	Mathematics	575	N502	3:00 P.M.

TABLE 8

Airline	*Flight Number*	*Gate*	*Destination*	*Departure Time*
Nadir	122	34	Detroit	08:10
Acme	221	22	Denver	08:17
Acme	122	33	Anchorage	08:22
Acme	323	34	Honolulu	08:30
Nadir	199	13	Detroit	08:47
Acme	222	22	Denver	09:10
Nadir	322	34	Detroit	09:44

Exercises

1. List the triples in the relation $\{(a, b, c) \mid a, b,$ and c are integers with $0 < a < b < c < 5\}$.
2. Which 4-tuples are in the relation $\{(a, b, c, d) \mid a, b, c,$ and d are positive integers with $abcd = 6\}$?
3. List the 5-tuples in the relation in Table 8.
4. Assuming that no new n-tuples are added, find all the primary keys for the relations displayed in
   a) Table 3.
   b) Table 5.
   c) Table 6.
   d) Table 8.
5. Assuming that no new n-tuples are added, find a composite key with two fields containing the *Airline* field for the database in Table 8.
6. What do you obtain when you apply the projection $P_{2,3,5}$ to the 5-tuple (a, b, c, d, e)?
7. Which projection mapping is used to delete the first, second, and fourth components of a 6-tuple?
8. Display the table produced by applying the projection $P_{1,2,4}$ to Table 8.
9. Display the table produced by applying the projection $P_{1,4}$ to Table 8.
10. How many components are there in the n-tuples in the table obtained by applying the join operator J_3 to two tables with 5-tuples and 8-tuples, respectively?
11. Construct the table obtained by applying the join operator J_2 to the relations in Tables 9 and 10.

TABLE 9

Supplier	*Part Number*	*Project*
23	1092	1
23	1101	3
23	9048	4
31	4975	3
31	3477	2
32	6984	4
32	9191	2
33	1001	1

TABLE 10

Part Number	*Project*	*Quantity*	*Color Code*
1001	1	14	8
1092	1	2	2
1101	3	1	1
3477	2	25	2
4975	3	6	2
6984	4	10	1
9048	4	12	2
9191	2	80	4

6.3

Representing Relations

INTRODUCTION

There are many ways to represent a relation between finite sets. As we have seen, one way is to list its ordered pairs. In this section we will discuss two alternative methods for representing relations. One method uses zero–one matrices. The other method uses directed graphs.

REPRESENTING RELATIONS USING MATRICES

A relation between finite sets can be represented using a zero–one matrix. Suppose that R is a relation from $A = \{a_1, a_2, \ldots, a_m\}$ to $B = \{b_1, b_2, \ldots, b_n\}$. (Here the elements of the sets A and B have been listed in a particular, but arbitrary, order. Furthermore, when $A = B$ we use the same ordering for A and B.) The relation R can be represented by the matrix $\mathbf{M}_R = [m_{ij}]$, where

$$m_{ij} = \begin{cases} 1 & \text{if } (a_i, b_j) \in R, \\ 0 & \text{if } (a_i, b_j) \notin R. \end{cases}$$

In other words, the zero–one matrix representing R has a 1 as its (i, j) entry when a_i is related to b_j, and a 0 in this position if a_i is not related to b_j. (Such a representation depends on the orderings used for A and B.)

The use of matrices to represent relations is illustrated in the following examples.

EXAMPLE 1 Suppose that $A = \{1, 2, 3\}$ and $B = \{1, 2\}$. Let R be the relation from A to B containing (a, b) if $a \in A$, $b \in B$, and $a > b$. What is the matrix representing R if $a_1 = 1$, $a_2 = 2$, and $a_3 = 3$, and $b_1 = 1$ and $b_2 = 2$?

Solution: Since $R = \{(2, 1), (3, 1), (3, 2)\}$, the matrix for R is

$$\mathbf{M}_R = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}.$$

The 1s in $\mathbf{M}_R$ show that the pairs (2, 1), (3, 1), and (3, 2) belong to R. The 0s show that no other pairs belong to R. ■

EXAMPLE 2 Let $A = \{a_1, a_2, a_3\}$ and $B = \{b_1, b_2, b_3, b_4, b_5\}$. Which ordered pairs are in the relation R represented by the matrix

$$\mathbf{M}_R = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}?$$

Solution: Since R consists of those ordered pairs (a_i, b_j) with $m_{ij} = 1$, it follows that

$$R = \{(a_1, b_2), (a_2, b_1), (a_2, b_3), (a_2, b_4), (a_3, b_1), (a_3, b_3), (a_3, b_5)\}.$$ ■

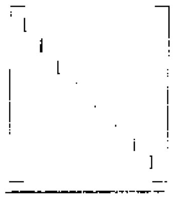


FIGURE 1 The Zero–One Matrix for a Reflexive Relation.

The matrix of a relation on a set, which is a square matrix, can be used to determine whether the relation has certain properties. Recall that a relation R on A is reflexive if $(a, a) \in R$ whenever $a \in A$. Thus, R is reflexive if and only if $(a_i, a_i) \in R$ for $i = 1, 2, \ldots, n$. Hence, R is reflexive if and only if $m_{ii} = 1$, for $i = 1, 2, \ldots, n$. In other words, R is reflexive if all the elements on the main diagonal of $\mathbf{M}_R$ are equal to 1, as shown in Figure 1.

The relation R is symmetric if $(a, b) \in R$ implies that $(b, a) \in R$. Consequently, the relation R on the set $A = \{a_1, a_2, \ldots, a_n\}$ is symmetric if and only if $(a_j, a_i) \in R$ whenever $(a_i, a_j) \in R$. In terms of the entries of $\mathbf{M}_R$, R is symmetric if and only if $m_{ji} = 1$ whenever $m_{ij} = 1$. This also means $m_{ji} = 0$ whenever $m_{ij} = 0$. Consequently, R is symmetric if and only if $m_{ij} = m_{ji}$, for all pairs of integers i and j with $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$. Recalling the definition of the transpose of a matrix from Section 2.6, we see that R is symmetric if and only if

$$\mathbf{M}_R = (\mathbf{M}_R)^t,$$

that is, if $\mathbf{M}_R$ is a symmetric matrix. The form of the matrix for a symmetric relation is illustrated in Figure 2(a).

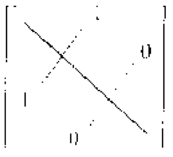
The relation R is antisymmetric if and only if $(a, b) \in R$ and $(b, a) \in R$ imply that $a = b$. Consequently, the matrix of an antisymmetric relation has the property that if $m_{ij} = 1$ with $i \neq j$, then $m_{ji} = 0$. Or, in other words, either $m_{ij} = 0$ or $m_{ji} = 0$ when $i \neq j$. The form of the matrix for an antisymmetric relation is illustrated in Figure 2(b).

EXAMPLE 3 Suppose that the relation R on a set is represented by the matrix

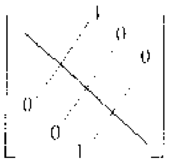
$$\mathbf{M}_R = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

Is R reflexive, symmetric, and/or antisymmetric?

Solution: Since all the diagonal elements of this matrix are equal to 1, R is reflexive. Moreover, since $\mathbf{M}_R$ is symmetric, it follows that R is symmetric. It is also easy to see that R is not antisymmetric. ■



(a) Symmetric



(b) Antisymmetric

FIGURE 2 The Zero–One Matrices for Symmetric and Antisymmetric Relations.

The Boolean operations join and meet (discussed in Section 2.6) can be used to find the matrices representing the union and the intersection of two relations. Suppose that R_1 and R_2 are relations on a set A represented by the matrices $\mathbf{M}_{R_1}$ and $\mathbf{M}_{R_2}$, respectively. The matrix representing the union of these relations has a 1 in the positions where either $\mathbf{M}_{R_1}$ or $\mathbf{M}_{R_2}$ has a 1. The matrix representing the intersection of these relations has a 1 in the positions where both $\mathbf{M}_{R_1}$ and $\mathbf{M}_{R_2}$ have a 1. Thus, the matrices representing the union and intersection of these relations are

$$\mathbf{M}_{R_1 \cup R_2} = \mathbf{M}_{R_1} \vee \mathbf{M}_{R_2} \quad \text{and} \quad \mathbf{M}_{R_1 \cap R_2} = \mathbf{M}_{R_1} \wedge \mathbf{M}_{R_2}.$$

EXAMPLE 4 Suppose that the relations R_1 and R_2 on a set A are represented by the matrices

$$\mathbf{M}_{R_1} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{M}_{R_2} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

What are the matrices representing $R_1 \cup R_2$ and $R_1 \cap R_2$?

Solution: The matrices of these relations are

$$\mathbf{M}_{R_1 \cup R_2} = \mathbf{M}_{R_1} \vee \mathbf{M}_{R_2} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix},$$

$$\mathbf{M}_{R_1 \cap R_2} = \mathbf{M}_{R_1} \wedge \mathbf{M}_{R_2} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

■

We now turn our attention to determining the matrix for the composite of relations. This matrix can be found using the Boolean product of the matrices (discussed in Section 2.6) for these relations. In particular, suppose that R is a relation from A to B and S is a relation from B to C. Suppose that A, B, and C have m, n, and p elements, respectively. Let the zero–one matrices for $S \circ R$, R, and S be $\mathbf{M}_{S \circ R} = [t_{ij}]$, $\mathbf{M}_R = [r_{ij}]$, and $\mathbf{M}_S = [s_{ij}]$, respectively (these matrices have sizes $m \times p$, $m \times n$, and $n \times p$, respectively). The ordered pair (a_i, c_j) belongs to $S \circ R$ if and only if there is an element b_k such that (a_i, b_k) belongs to R and (b_k, c_j) belongs to S. It follows that $t_{ij} = 1$ if and only if $r_{ik} = s_{kj} = 1$ for some k. From the definition of the Boolean product, this means that

$$\mathbf{M}_{S \circ R} = \mathbf{M}_R \odot \mathbf{M}_S.$$

EXAMPLE 5 Find the matrix representing the relations $S \circ R$ where the matrices representing R and S are

$$\mathbf{M}_R = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{M}_S = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}.$$

Solution: The matrix for $S \circ R$ is

$$\mathbf{M}_{S \circ R} = \mathbf{M}_R \odot \mathbf{M}_S = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

■

The matrix representing the composite of two relations can be used to find the matrix for $\mathbf{M}_{R^n}$. In particular,

$$\mathbf{M}_{R^n} = \mathbf{M}_R^{[n]},$$

from the definition of Boolean powers. Exercise 19 at the end of this section asks for a proof of this formula.

EXAMPLE 6 Find the matrix representing the relation R^2 where the matrix representing R is

$$\mathbf{M}_R = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

Solution: The matrix for R^2 is

$$\mathbf{M}_{R^2} = \mathbf{M}_R^{[2]} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

■

REPRESENTING RELATIONS USING DIGRAPHS

We have shown that a relation can be represented by listing all of its ordered pairs or by using a zero–one matrix. There is another important way of representing a relation using a pictorial representation. Each element of the set is represented by a point, and each ordered pair is represented using an arc with its direction indicated by an arrow. We use such pictorial representations when we think of relations on a finite set as **directed graphs,** or **digraphs.**

DEFINITION 1. **A *directed graph,* or *digraph,* consists of a set V of *vertices* (or *nodes*) together with a set E of ordered pairs of elements of V called *edges* (or *arcs*). The vertex a is called the *initial vertex* of the edge (a, b), and the vertex b is called the *terminal vertex* of this edge.**

An edge of the form (a, a) is represented using an arc from the vertex a back to itself. Such an edge is called a **loop.**

EXAMPLE 7 The directed graph with vertices a, b, c, and d, and edges (a, b), (a, d), (b, b), (b, d), (c, a), (c, b), and (d, b) is displayed in Figure 3. ■

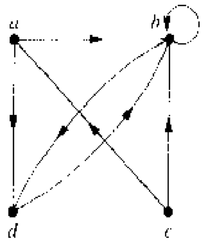


FIGURE 3 A Directed Graph.

The relation R on a set A is represented by the directed graph that has the elements of A as its vertices and the ordered pairs (a, b), where $(a, b) \in R$, as edges. This assignment sets up a one-to-one correspondence between the relations on a set A and the directed graphs with A as their set of vertices. Thus, every statement about relations corresponds to a statement about directed graphs, and vice versa. Directed graphs give a visual display of information about relations. As such, they are often used to study relations and their properties. (Note that relations from a set A to a set B can be represented by a directed graph where there is a vertex for each element of A and a vertex for each element of B as shown in Section 6.1. However, when $A = B$, such representation provides much less insight than the digraph representations described here.) The use of directed graphs to represent relations is illustrated in the following examples.

EXAMPLE 8 The directed graph of the relation

$$R = \{(1,1),(1,3),(2,1),(2,3),(2,4),(3,1),(3,2),(4,1)\}$$

on the set $\{1,2,3,4\}$ is shown in Figure 4. ■

EXAMPLE 9 What are the ordered pairs in the relation R represented by the directed graph shown in Figure 5?

Solution: The ordered pairs (x, y) in the relation are

$$R = \{(1,3),(1,4),(2,1),(2,2),(2,3),(3,1),(3,3),(4,1),(4,3)\}.$$

Each of these pairs corresponds to an edge of the directed graph, with $(2, 2)$ and $(3, 3)$ corresponding to loops. ■

The directed graph representing a relation can be used to determine whether the relation has various properties. For instance, a relation is reflexive if and only if there is a loop at every vertex of the directed graph, so that every ordered pair of the form (x, x) occurs in the relation. A relation is symmetric if and only if for every edge between distinct vertices in its digraph there is an edge in the opposite direction, so that (y, x) is in the relation whenever (x, y) is in the relation. Similarly, a relation is antisymmetric if and only if there are never two edges in opposite directions between distinct vertices. Finally, a relation is transitive if and only if whenever there is an edge from a vertex x to a vertex y and an edge from a vertex y to a vertex z, there is an edge from x to z (completing a triangle where each side is a directed edge with the correct direction).

EXAMPLE 10 Determine whether the relations for the directed graphs shown in Figure 6 are reflexive, symmetric, antisymmetric, and/or transitive.

Solution: Since there are loops at every vertex of the directed graph of R, it is reflexive. R is neither symmetric nor antisymmetric since there is an edge from a to b but not one from b to a, but there are edges in both directions connecting b and c. Finally, R is not transitive since there is an edge from a to b and an edge from b to c, but no edge from a to c.

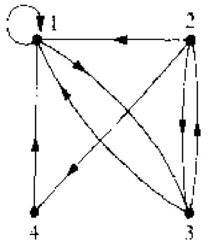


FIGURE 4 The Directed Graph of the Relation R.

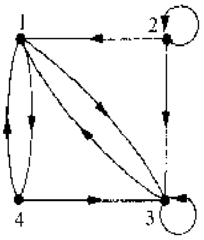
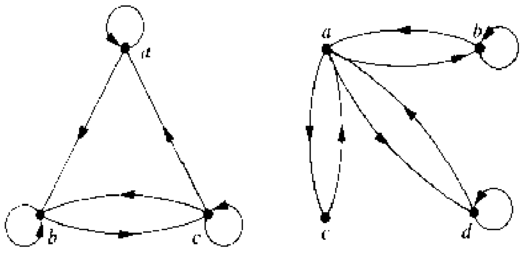


FIGURE 5 The Directed Graph of the Relation R.



(a) Directed graph of R (b) Directed graph of S

FIGURE 6 **The Directed Graphs of the Relations R ans S.**

Since loops are not present at all the vertices of the directed graph of S, this relation is not reflexive. It is symmetric and not antisymmetric, since every edge between distinct vertices is accompanied by an edge in the opposite direction. It is also not hard to see from the directed graph that S is not transitive, since (c, a) and (a, b) belong to S, but (c, b) does not belong to S. ■

Exercises

1. Represent each of the following relations on $\{1, 2, 3\}$ with a matrix (with the elements of this set listed in increasing order).

a) $\{(1, 1), (1, 2), (1, 3)\}$
b) $\{(1, 2), (2, 1), (2, 2), (3, 3)\}$
c) $\{(1, 1), (1, 2), (1, 3), (2, 2), (2, 3), (3, 3)\}$
d) $\{(1, 3), (3, 1)\}$

2. List the ordered pairs in the relations on $\{1, 2, 3\}$ corresponding to the following matrices (where the rows and columns correspond to the integers listed in increasing order).

a) $\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$ **b)** $\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$ **c)** $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$

3. How can the matrix for a relation be used to determine whether the relation is irreflexive?

4. Determine whether the relations represented by the matrices in Exercise 2 are reflexive, irreflexive, symmetric, antisymmetric, and/or transitive.

5. How can the matrix for $\overline{R}$, the complement of the relation R, be found from the matrix representing R, when R is a relation on a finite set A?

6. How can the matrix for R^{-1}, the inverse of the relation R, be found from the matrix representing R, when R is a relation on a finite set A?

7. Let R be the relation represented by the matrix

$$\mathbf{M}_R = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

Find the matrix representing

a) R^{-1}. **b)** $\overline{R}$. **c)** R^2.

8. Let R_1 and R_2 be relations on a set A represented by the matrices

$$\mathbf{M}_{R_1} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \text{ and } \mathbf{M}_{R_2} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Find the matrices that represent

a) $R_1 \cup R_2$. **b)** $R_1 \cap R_2$. **c)** $R_2 \circ R_1$.
d) $R_1 \circ R_1$. **e)** $R_1 \oplus R_2$.

9. Let R be the relation represented by the matrix

$$\mathbf{M}_R = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

Find the matrices that represent

a) R^2. **b)** R^3. **c)** R^4.

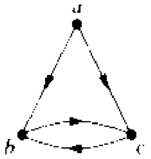
10. Draw the directed graphs representing each of the relations from Exercise 1.

11. Draw the directed graphs representing each of the relations from Exercise 2.

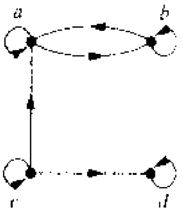
12. Draw the directed graph that represents the relation $\{(a, a), (a, b), (b, c), (c, b), (c, d), (d, a), (d, b)\}$.

In Exercises 13–15 list the ordered pairs in the relations represented by the directed graphs.

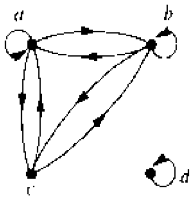
13.



14.



15.



16. How can the directed graph of a relation R on a finite set A be used to determine if a relation is irreflexive?

17. Determine whether the relations represented by the directed graphs shown in Exercises 13–15 are reflexive, irreflexive, symmetric, antisymmetric, and/or transitive.

18. Given the directed graphs representing two relations, how can the directed graph of the union, intersection, symmetric difference, difference, and composition of these relations be found?

19. Show that if $\mathbf{M}_R$ is the matrix representing the relation R, then $\mathbf{M}_R^{[n]}$ is the matrix representing the relation R^n.

6.4

Closures of Relations

INTRODUCTION

A computer network has data centers in Boston, Chicago, Denver, Detroit, New York, and San Diego. There are direct, one-way telephone lines from Boston to Chicago, from Boston to Detroit, from Chicago to Detroit, from Detroit to Denver, and from New York to San Diego. Let R be the relation containing (a, b) if there is a telephone line from the data center in a to that in b. How can we determine if there is some (possibly indirect) link composed of one or more telephone lines from one center to another? Since not all links are direct, such as the link from Boston to Denver that goes through Detroit, R cannot be used directly to answer this. In the language of relations, R is not transitive, so it does not contain all the pairs that can be linked. As we will show in this section, we can find all pairs of data centers that have a link by constructing the smallest transitive relation that contains R. This relation is called the **transitive closure** of R.

In general, let R be a relation on a set A. R may or may not have some property **P**, such as reflexivity, symmetry, or transitivity. If there is a relation S with property **P** containing R such that S is a subset of every relation with property **P** containing R, then S is called the **closure** of R with respect to **P**. (Note that the closure of a relation with respect to a property may not exist; see Exercises 15 and 35 at the end of this section.) We will show how reflexive, symmetric, and transitive closures of relations can be found.

CLOSURES

The relation $R = \{(1, 1), (1, 2), (2, 1), (3, 2)\}$ on the set $A = \{1, 2, 3\}$ is not reflexive. How can we produce a reflexive relation containing R that is as small as possible? This can be done by adding (2, 2) and (3, 3) to R, since these are the only pairs of the form (a, a) that are not in R. Clearly, this new relation contains R. Furthermore, *any* reflexive relation that contains R must also contain (2, 2) and (3, 3). Because this relation contains R, is reflexive, and is contained within every reflexive relation that contains R, it is called the **reflexive closure** of R.

As this example illustrates, given a relation R on a set A, the reflexive closure of R can be formed by adding to R all pairs of the form (a, a) with $a \in A$, not already in R. The addition of these pairs produces a new relation that is reflexive, contains R, and is contained within any reflexive relation containing R. We see that the reflexive closure of R equals $R \cup \Delta$, where $\Delta = \{(a, a) \mid a \in A\}$ is the **diagonal relation** on A. (The reader should verify this.)

EXAMPLE 1 What is the reflexive closure of the relation $R = \{(a, b) \mid a < b\}$ on the set of integers?

Solution: The reflexive closure of R is

$$R \cup \Delta = \{(a, b) \mid a < b\} \cup \{(a, a) \mid a \in \mathbf{Z}\} = \{(a, b) \mid a \leq b\}. \qquad \blacksquare$$

The relation $\{(1, 1), (1, 2), (2, 2), (2, 3), (3, 1), (3, 2)\}$ on $\{1, 2, 3\}$ is not symmetric. How can we produce a symmetric relation that is as small as possible and contains R? To do this, we need only add (2, 1) and (1, 3), since these are the only pairs of the form (b, a) with $(a, b) \in R$ that are not in R. This new relation is symmetric and contains R. Furthermore, *any* symmetric relation that contains R must contain this new relation, since a symmetric relation that contains R must contain (2, 1) and (1, 3). Consequently, this new relation is called the **symmetric closure** of R.

As this example illustrates, the symmetric closure of a relation R can be constructed by adding all ordered pairs of the form (b, a), where (a, b) is in the relation, that are not already present in R. Adding these pairs produces a relation that is symmetric, that contains R, and that is contained in any symmetric relation that contains R. The symmetric closure of a relation can be constructed by taking the union of a relation with its inverse; that is, $R \cup R^{-1}$ is the symmetric closure of R, where $R^{-1} = \{(b, a) \mid (a, b) \in R\}$. The reader should verify this statement.

EXAMPLE 2 What is the symmetric closure of the relation $R = \{(a, b) \mid a > b\}$ on the set of positive integers?

Solution: The symmetric closure of R is the relation

$$R \cup R^{-1} = \{(a, b) \mid a > b\} \cup \{(b, a) \mid a > b\} = \{(a, b) \mid a \neq b\}. \qquad \blacksquare$$

Suppose that a relation R is not transitive. How can we produce a transitive relation that contains R such that this new relation is contained within any transitive relation that contains R? Can the transitive closure of a relation R be produced by adding all the pairs of the form (a, c) where (a, b) and (b, c) are already in the relation? Consider the relation $R = \{(1, 3), (1, 4), (2, 1), (3, 2)\}$ on the set $\{1, 2, 3, 4\}$. This relation is not transitive since it does not contain all pairs of the form (a, c) where (a, b) and (b, c) are in R. The pairs of this form not in R are (1, 2), (2, 3), (2, 4), and (3, 1). Adding these pairs does *not* produce a transitive relation, since the resulting relation contains (3, 1) and (1, 4) but does not contain (3, 4). This shows that constructing the transitive closure of a relation is more complicated than constructing either the reflexive or symmetric closure. The rest of this section develops algorithms for constructing transitive closures. As will be shown, the transitive closure of a relation can be found by adding new ordered pairs that must be present and then repeating this process until no new ordered pairs are needed.

PATHS IN DIRECTED GRAPHS

We will see that representing relations by directed graphs helps in the construction of transitive closures. We now introduce some terminology that we will use for this purpose.

A path in a directed graph is obtained by traversing along edges (in the same direction as indicated by the arrow on the edge).

DEFINITION 1. A *path* from a to b in the directed graph G is a sequence of one or more edges $(x_0, x_1), (x_1, x_2), (x_2, x_3), \ldots, (x_{n-1}, x_n)$ in G, where $x_0 = a$ and $x_n = b$, that is, a sequence of edges where the terminal vertex of an edge is the same as the initial vertex in the next edge in the path. This path is denoted by $x_0, x_1, x_2, \ldots, x_{n-1}, x_n$ and has *length* n. A path that begins and ends at the same vertex is called a *circuit* or *cycle*.

A path in a directed graph can pass through a vertex more than once. Moreover, an edge in a directed graph can occur more than once in a path. The reader should note that some authors allow paths of length zero, that is, paths consisting of no edges. In this book all paths must have a length of at least one.

EXAMPLE 3 Which of the following are paths in the directed graph shown in Figure 1: a, b, e, d; a, e, c, d, b; b, a, c, b, a, a, b; d, c; c, b, a; e, b, a, b, a, b, e? What are the lengths of those that are paths? Which of the paths in this list are circuits?

Solution: Since each of (a, b), (b, e), and (e, d) is an edge, a, b, e, d is a path of length three. Since (c, d) is not an edge, a, e, c, d, b is not a path. Also, b, a, c, b, a, a, b is a path of length six since (b, a), (a, c), (c, b), (b, a), (a, a), and (a, b) are all edges. We see that d, c is a path of length one, since (d, c) is an edge. Also c, b, a is a path of length two.

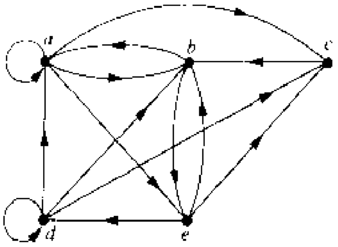


FIGURE 1 A Directed Graph.

since (c, b) and (b, a) are edges. All of (e, b), (b, a), (a, b), (b, a), (a, b), and (b, e) are edges, so that e, b, a, b, a, b, e is a path of length six.

The two paths b, a, c, b, a, a, b and e, b, a, b, a, b, e are circuits since they begin and end at the same vertex. The paths a, b, e, d; c, b, a; and d, c are not circuits. ■

The term *path* also applies to relations. Carrying over the definition from directed graphs to relations, there is a **path** from a to b in R if there is a sequence of elements $a, x_1, x_2, \ldots, x_{n-1}, b$ with $(a, x_1) \in R$, $(x_1, x_2) \in R, \ldots,$ and $(x_{n-1}, b) \in R$. The following theorem can be obtained from the definition of a path in a relation.

THEOREM 1 **Let R be a relation on a set A. There is a path of length n from a to b if and only if $(a, b) \in R^n$.**

Proof: We will use mathematical induction. By definition, there is a path from a to b of length one if and only if $(a, b) \in R$, so the theorem is true when $n = 1$.

Assume that the theorem is true for the positive integer n. This is the inductive hypothesis. There is a path of length $n + 1$ from a to b if and only if there is an element $c \in A$ such that there is a path of length one from a to c, so $(a, c) \in R$, and a path of length n from c to b, that is, $(c, b) \in R^n$. Consequently, by the induction hypothesis, there is a path of length $n + 1$ from a to b if and only if there is an element c with $(a, c) \in R$ and $(c, b) \in R^n$. But there is such an element if and only if $(a, b) \in R^{n+1}$. Therefore, there is a path of length $n + 1$ from a to b if and only if $(a, b) \in R^{n+1}$. This completes the proof. □

TRANSITIVE CLOSURES

We now show that finding the transitive closure of a relation is equivalent to determining which pairs of vertices in the associated directed graph are connected by a path. With this in mind, we define a new relation.

DEFINITION 2. Let R be a relation on a set A. The *connectivity relation* R^* consists of the pairs (a, b) such that there is a path between a and b in R.

Since R^n consists of the pairs (a, b) such that there is a path of length n from a to b, it follows that R^* is the union of all the sets R^n. In other words,

$$R^* = \bigcup_{n=1}^{\infty} R^n.$$

The connectivity relation is useful in many models.

EXAMPLE 4 Let R be the relation on the set of all people in the world that contains (a, b) if a has met b. What is R^n, where n is a positive integer greater than 2? What is R^*?

Solution: The relation R^2 contains (a, b) if there is a person c such that $(a, c) \in R$ and $(c, b) \in R$, that is, if there is a person c such that a has met c and c has met b. Similarly,

R^n consists of those pairs (a, b) such that there are people $x_1, x_2, \ldots, x_{n-1}$ such that a has met x_1, x_1 has met $x_2, \ldots,$ and x_{n-1} has met b.

The relation R^* contains (a, b) if there is a sequence of people, starting with a and ending with b, such that each person in the sequence has met the next person in the sequence. (There are many interesting conjectures about R^*. Do you think that this connectivity relation includes the pair with you as the first element and the president of Mongolia as the second element?) ■

EXAMPLE 5 Let R be the relation on the set of all subway stops in New York City that contains (a, b) if it is possible to travel from stop a to stop b without changing trains. What is R^n when n is a positive integer? What is R^*?

Solution: The relation R^n contains (a, b) if it is possible to travel from stop a to stop b by making at most $n - 1$ changes of trains. The relation R^* consists of the ordered pairs (a, b) where it is possible to travel from stop a to stop b making as many changes of trains as necessary. (The reader should verify these statements.) ■

EXAMPLE 6 Let R be the relation on the set of all states in the United States that contains (a, b) if state a and state b have a common border. What is R^n where n is a positive integer? What is R^*?

Solution: The relation R^n consists of the pairs (a, b) where it is possible to go from state a to state b by crossing exactly n state borders. R^* consists of the ordered pairs (a, b) where it is possible to go from state a to state b crossing as many borders as necessary. (The reader should verify these statements.) The only ordered pairs not in R^* are those containing states that are not connected to the continental United States (i.e., those pairs containing Alaska or Hawaii). ■

The following theorem shows that the transitive closure of a relation and the associated connectivity relation are the same.

THEOREM 2 **The transitive closure of a relation R equals the connectivity relation R^*.**

Proof: Note that R^* contains R. To show that R^* is the transitive closure of R we must also show that R^* is transitive and that $R^* \subseteq S$ whenever S is a transitive relation that contains R.

First, we show that R^* is transitive. If $(a, b) \in R^*$ and $(b, c) \in R^*$, then there are paths from a to b and from b to c in R. We obtain a path from a to c by starting with the path from a to b and following it with the path from b to c. Hence, $(a, c) \in R^*$. It follows that R^* is transitive.

Now suppose that S is a transitive relation containing R. Since S is transitive, S^n also is transitive (the reader should verify this) and $S^n \subseteq S$ (by Theorem 1 of Section 6.1). Furthermore, since

$$S^* = \bigcup_{k=1}^{\infty} S^k,$$

and $S^k \subseteq S$, it follows that $S^* \subseteq S$. Now note that if $R \subseteq S$, then $R^* \subseteq S^*$, because any path in R is also a path in S. Consequently, $R^* \subseteq S^* \subseteq S$. Thus, any transitive relation that contains R must also contain R^*. Therefore, R^* is the transitive closure of R. ☐

Now that we know that the transitive closure equals the connectivity relation, we turn our attention to the problem of computing this relation. We do not need to examine arbitrarily long paths to determine whether there is a path between two vertices in a finite directed graph. As the following lemma shows, it is sufficient to examine paths containing no more than n edges, where n is the number of elements in the set.

LEMMA 1 **Let A be a set with n elements, and let R be a relation on A. If there is a path in R from a to b, then there is such a path with length not exceeding n. Moreover, when $a \neq b$, if there is a path in R from a to b, then there is such a path with length not exceeding $n - 1$.**

Proof: Suppose there is a path from a to b in R. Let m be the length of the shortest such path. Suppose that $x_0, x_1, x_2, \ldots, x_{m-1}, x_m$, where $x_0 = a$ and $x_m = b$, is such a path.

Suppose that $a = b$ and that $m > n$, so that $m \geq n+1$. By the pigeonhole principle, since there are n vertices in A, among the m vertices $x_0, x_1, \ldots, x_{m-1}$, at least two are equal (see Figure 2).

Suppose that $x_i = x_j$ with $0 \leq i < j \leq m - 1$. Then the path contains a circuit from x_i to itself. This circuit can be deleted from the path from a to b, leaving a path, namely, $x_0, x_1, \ldots, x_i, x_{j+1}, \ldots, x_{m-1}, x_m$, from a to b of shorter length. Hence, the path of shortest length must have length less than or equal to n.

The case where $a \neq b$ is left as an exercise for the reader. ☐

From Lemma 1, we see that the transitive closure of R is the union of R, R^2, $R^3, \ldots$, and R^n. This follows since there is a path in R^* between two vertices if and only if there is a path between these vertices in R^i, for some positive integer i with $i \leq n$. Since

$$R^* = R \cup R^2 \cup R^3 \cup \cdots \cup R^n,$$

and the zero–one matrix representing a union of relations is the join of the zero–one matrices of these relations, the zero–one matrix for the transitive closure is the join of the zero–one matrices of the first n powers of the zero–one matrix of R.

THEOREM 3 **Let $\mathbf{M}_R$ be the zero–one matrix of the relation R on a set with n elements. Then the zero–one matrix of the transitive closure R^* is**

$$\mathbf{M}_{R^*} = \mathbf{M}_R \vee \mathbf{M}_R^{[2]} \vee \mathbf{M}_R^{[3]} \vee \cdots \vee \mathbf{M}_R^{[n]}.$$

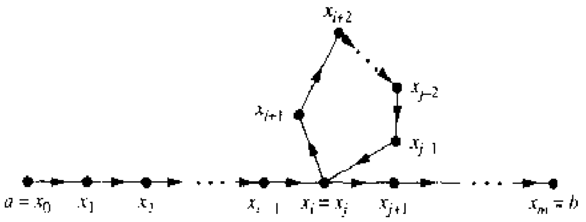


FIGURE 2 **Producing a Path with Length Not Exceeding n.**

EXAMPLE 7 Find the zero–one matrix of the transitive closure of the relation R where

$$\mathbf{M}_R = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}.$$

Solution: From Theorem 3, it follows that the zero–one matrix of R^* is

$$\mathbf{M}_{R^*} = \mathbf{M}_R \vee \mathbf{M}_R^{[2]} \vee \mathbf{M}_R^{[3]}.$$

Since

$$\mathbf{M}_R^{[2]} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{M}_R^{[3]} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix},$$

it follows that

$$\mathbf{M}_{R^*} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} \vee \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \vee \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}.$$

■

Theorem 3 can be used as a basis for an algorithm for computing the matrix of the relation R^*. To find this matrix, the successive Boolean powers of $\mathbf{M}_R$, up to the nth power, are computed. As each power is calculated, its join with the join of all smaller powers is formed. When this is done with the nth power, the matrix for R^* has been found. This procedure is displayed as Algorithm 1.

ALGORITHM 1 **A Procedure for Computing the Transitive Closure.**

```
procedure transitive closure ($\mathbf{M}_R$ : zero–one $n \times n$ matrix)
A := $\mathbf{M}_R$
B := A
for $i$ := 2 to $n$
begin
    A := A $\odot$ $\mathbf{M}_R$
    B := B $\vee$ A
end {$B$ is the zero–one matrix for $R^*$}
```

We can easily find the number of bit operations used by Algorithm 1 to determine the transitive closure of a relation. Computing the Boolean powers $\mathbf{M}_R, \mathbf{M}_R^{[2]}, \ldots, \mathbf{M}_R^{[n]}$ requires that $n - 1$ Boolean products of $n \times n$ zero–one matrices be found. Each of these Boolean products can be found using $n^2(2n - 1)$ bit operations. Hence, these products can be computed using $n^2(2n - 1)(n - 1)$ bit operations.

To find $\mathbf{M}_{R^*}$ from the n Boolean powers of $\mathbf{M}_R$, $n - 1$ joins of zero–one matrices need to be found. Computing each of these joins uses n^2 bit operations. Hence, $(n - 1)n^2$ bit operations are used in this part of the computation. Therefore, when Algorithm 1 is used, the matrix of the transitive closure of a relation on a set with n elements can be found using $n^2(2n - 1)(n - 1) + (n - 1)n^2 = 2n^3(n - 1) = O(n^4)$ bit operations. The remainder of this section describes a more efficient algorithm for finding transitive closures.

WARSHALL'S ALGORITHM

Warshall's algorithm, named after Stephen Warshall, who described this algorithm in 1960, is an efficient method for computing the transitive closure of a relation. Algorithm 1 can find the transitive closure of a relation on a set with n elements using $2n^3(n-1)$ bit operations. However, the transitive closure can be found by Warshall's algorithm using only $2n^3$ bit operations.

Remark: Warshall's algorithm is sometimes called the Roy–Warshall algorithm, since B. Roy described this algorithm in 1959.

Suppose that R is a relation on a set with n elements. Let $v_1, v_2, \ldots, v_n$ be an arbitrary listing of these n elements. The concept of the **interior vertices** of a path is used in Warshall's algorithm. If $a, x_1, x_2, \ldots, x_{m-1}, b$ is a path, its interior vertices are $x_1, x_2, \ldots, x_{m-1}$, that is, all the vertices of the path that occur somewhere other than as the first and last vertices in the path. For instance, the interior vertices of a path a, c, d, f, g, h, b, j in a directed graph are c, d, f, g, h, and b. The interior vertices of a, c, d, a, f, b are c, d, a, and f. (Note that the first vertex in the path is not an interior vertex unless it is visited again by the path, except as the last vertex. Similarly, the last vertex in the path is not an interior vertex unless it was visited previously by the path, except as the first vertex.)

Warshall's algorithm is based on the construction of a sequence of zero–one matrices. These matrices are $\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_n$, where $\mathbf{W}_0 = \mathbf{M}_R$ is the zero–one matrix of this relation, and $\mathbf{W}_k = [w_{ij}^{(k)}]$, where $w_{ij}^{(k)} = 1$ if there is a path from v_i to v_j such that all the interior vertices of this path are in the set $\{v_1, v_2, \ldots, v_k\}$ (the first k vertices in the list) and is 0 otherwise. (The first and last vertices in the path may be outside the set of the first k vertices in the list.) Note that $\mathbf{W}_n = \mathbf{M}_{R^*}$, since the (i, j)th entry of $\mathbf{M}_{R^*}$ is 1 if and only if there is a path from v_i to v_j, with all interior vertices in the set $\{v_1, v_2, \ldots, v_n\}$ (but these are the only vertices in the directed graph). The following example illustrates what the matrix $\mathbf{W}_k$ represents.

Stephen Warshall (born 1935). Stephen Warshall, born in New York City, went to public school in Brooklyn. He attended Harvard University, receiving his degree in mathematics in 1956. He never received an advanced degree, since at that time no programs were available in his areas of interest. However, he took graduate courses at several different universities and contributed to the development of computer science and software engineering.

After graduating from Harvard, Warshall worked at ORO (Operation Research Office), which was set up by Johns Hopkins to do research and development for the U.S. Army. In 1958 he left ORO to take a position at a company called Technical Operations, where he helped build a research and development laboratory for military software projects. In 1961 he left Technical Operations to found Massachusetts Computer Associates. Later, this company became part of Applied Data Research (ADR). After the merger, Warshall sat on the board of directors of ADR and managed a variety of projects and organizations. He retired from ADR in 1982.

During his career Warshall carried out research and development in operating systems, compiler design, language design, and operations research. In the 1971–1972 academic year he presented lectures on software engineering at French universities. There is an interesting anecdote about his proof that the transitive closure algorithm, now known as Warshall's algorithm, is correct. He and a colleague at Technical Operations bet a bottle of rum on who first could determine whether this algorithm always works. Warshall came up with his proof overnight, winning the bet and the rum, which he shared with the loser of the bet. Because Warshall did not like sitting at a desk, he did much of his creative work in unconventional places, such as on a sailboat in the Indian Ocean or in a Greek lemon orchard.

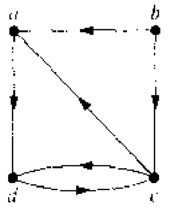


FIGURE 3 **The Directed Graph of the Relation R.**

EXAMPLE 8 Let R be the relation with directed graph shown in Figure 3. Let a, b, c, d be a listing of the elements of the set. Find the matrices $\mathbf{W}_0$, $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{W}_3$, and $\mathbf{W}_4$. The matrix $\mathbf{W}_4$ is the transitive closure of R.

Solution: Let $v_1 = a$, $v_2 = b$, $v_3 = c$, and $v_4 = d$. $\mathbf{W}_0$ is the matrix of the relation. Hence,

$$\mathbf{W}_0 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

W_1 has 1 as its (i, j)th entry if there is a path from v_i to v_j that has only $v_1 = a$ as an interior vertex. Note that all paths of length one can still be used since they have no interior vertices. Also, there is now an allowable path from b to d, namely, b, a, d. Hence,

$$\mathbf{W}_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

$\mathbf{W}_2$ has 1 as its (i, j)th entry if there is a path from v_i to v_j that has only $v_1 = a$ and/or $v_2 = b$ as its interior vertices, if any. Since there are no edges that have b as a terminal vertex, no new paths are obtained when we permit b to be an interior vertex. Hence, $\mathbf{W}_2 = \mathbf{W}_1$.

$\mathbf{W}_3$ has 1 as its (i, j)th entry if there is a path from v_i to v_j that has only $v_1 = a$, $v_2 = b$, and/or $v_3 = c$ as its interior vertices, if any. We now have paths from d to a, namely, d, c, a, and from d to d, namely, d, c, d. Hence,

$$\mathbf{W}_3 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}.$$

Finally, $\mathbf{W}_4$ has 1 as its (i, j)th entry if there is a path from v_i to v_j that has $v_1 = a$, $v_2 = b$, $v_3 = c$, and/or $v_4 = d$ as interior vertices, if any. Since these are all the vertices of the graph, this entry is 1 if and only if there is a path from v_i to v_j. Hence,

$$\mathbf{W}_4 = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}.$$

This last matrix, $\mathbf{W}_4$, is the matrix of the transitive closure. ■

Warshall's algorithm computes $\mathbf{M}_{R^*}$ by efficiently computing $\mathbf{W}_0 = \mathbf{M}_R, \mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_n = \mathbf{M}_{R^*}$. The following observation shows that we can compute $\mathbf{W}_k$ directly from $\mathbf{W}_{k-1}$: There is a path from v_i to v_j with no vertices other than $v_1, v_2, \ldots, v_k$ as interior vertices if and only if either there is a path from v_i to v_j with its interior vertices among the first $k-1$ vertices in the list, or there are paths from v_i to v_k and from v_k to v_j that have interior vertices only among the first $k-1$ vertices in the list. That is, either a path from v_i to v_j already existed before v_k was permitted as an interior vertex, or allowing v_k as an interior vertex produces a path that goes from v_i to v_k and then from v_k to v_j. These two cases are shown in Figure 4.

The first type of path exists if and only if $w_{ij}^{[k-1]} = 1$, and the second type of path exists if and only if both $w_{ik}^{[k-1]}$ and $w_{kj}^{[k-1]}$ are 1. Hence, $w_{ij}^{[k]}$ is 1 if and only if either $w_{ij}^{[k-1]}$ is 1 or both $w_{ik}^{[k-1]}$ and $w_{kj}^{[k-1]}$ are 1. This gives us the following lemma.

LEMMA 2 **Let $\mathbf{W}_k = [w_{ij}^{[k]}]$ be the zero–one matrix that has a 1 in its (i, j)th position if and only if there is a path from v_i to v_j with interior vertices from the set $\{v_1, v_2, \ldots, v_k\}$. Then**

$$w_{ij}^{[k]} = w_{ij}^{[k-1]} \vee (w_{ik}^{[k-1]} \wedge w_{kj}^{[k-1]}),$$

whenever i, j, and k are positive integers not exceeding n.

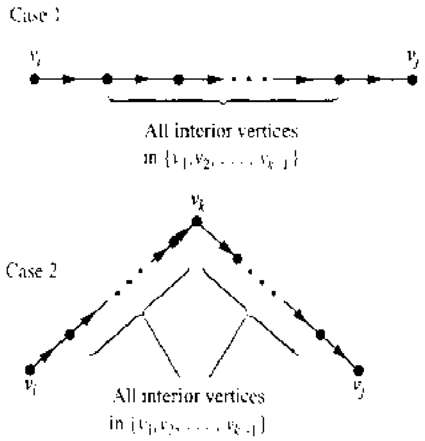

Case 1
v_i
v_j
All interior vertices in $\{v_1, v_2, \ldots, v_{k-1}\}$
v_k
Case 2
v_i
v_j
All interior vertices in $\{v_1, v_2, \ldots, v_{k-1}\}$

FIGURE 4 Adding v_k to the Set of Allowable Interior Vertices.

Lemma 2 gives us the means efficiently to compute the matrices $\mathbf{W}_k$, $k = 1, 2, \ldots, n$. We display the pseudocode for Warshall's algorithm, using Lemma 2, as Algorithm 2.

ALGORITHM 2 Warshall Algorithm.

```
procedure Warshall ($\mathbf{M}_R$ : $n \times n$ zero-one matrix)
$\mathbf{W} := \mathbf{M}_R$
for $k := 1$ to $n$
begin
    for $i := 1$ to $n$
    begin
        for $j := 1$ to $n$
            $w_{ij} := w_{ij} \vee (w_{ik} \wedge w_{kj})$
    end
end {$\mathbf{W} = [w_{ij}]$ is $\mathbf{M}_{R^*}$}
```

The computational complexity of Warshall's algorithm can easily be computed in terms of bit operations. To find the entry $w_{ij}^{[k]}$ from the entries $w_{ij}^{[k-1]}$, $w_{ik}^{[k-1]}$, and $w_{kj}^{[k-1]}$ using Lemma 2 requires two bit operations. To find all n^2 entries of $\mathbf{W}_k$ from those of $\mathbf{W}_{k-1}$ requires $2n^2$ bit operations. Since Warshall's algorithm begins with $\mathbf{W}_0 = \mathbf{M}_R$ and computes the sequence of n zero–one matrices $\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_n = \mathbf{M}_{R^*}$, the total number of bit operations used is $n \cdot 2n^2 = 2n^3$.

Exercises

1. Let R be the relation on the set $\{0, 1, 2, 3\}$ containing the ordered pairs $(0, 1)$, $(1, 1)$, $(1, 2)$, $(2, 0)$, $(2, 2)$, and $(3, 0)$. Find the

a) reflexive closure of R. **b)** symmetric closure of R.

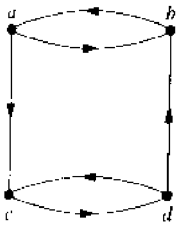
2. Let R be the relation $\{(a, b) \mid a \neq b\}$ on the set of integers. What is the reflexive closure of R?

3. Let R be the relation $\{(a, b) \mid a \text{ divides } b\}$ on the set of integers. What is the symmetric closure of R?

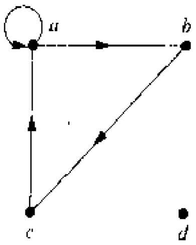
4. How can the directed graph representing the reflexive closure of a relation on a finite set be constructed from the directed graph of the relation?

In Exercises 5–7 draw the directed graph of the reflexive closure of the relations with the directed graph shown.

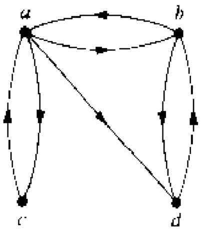
5.



6.



7.



8. How can the directed graph representing the symmetric closure of a relation on a finite set be constructed from the directed graph for this relation?

9. Find the directed graphs of the symmetric closures of the relations with directed graphs shown in Exercises 5–7.

10. Find the smallest relation containing the relation in Example 2 that is both reflexive and symmetric.

11. Find the directed graph of the smallest relation that is both reflexive and symmetric for each of the relations with directed graphs shown in Exercises 5–7.

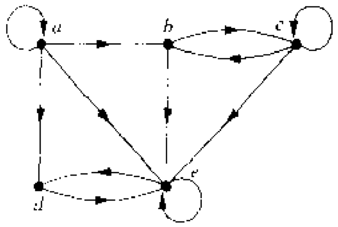
12. Suppose that the relation R on the finite set A is represented by the matrix $\mathbf{M}_R$. Show that the matrix that represents the reflexive closure of R is $\mathbf{M}_R \vee \mathbf{I}_n$.

13. Suppose that the relation R on the finite set A is represented by the matrix $\mathbf{M}_R$. Show that the matrix that represents the symmetric closure of R is $\mathbf{M}_R \vee \mathbf{M}_R^t$.

14. Show that the closure of a relation R with respect to a property **P**, if it exists, is the intersection of all the relations with property **P** that contain R.

15. When is it possible to define the "irreflexive closure" of a relation R, that is, a relation that contains R, is irreflexive, and is contained in every irreflexive relation that contains R?

16. Determine whether the following sequences of vertices are paths in the following directed graph.



   a) a, b, c, e
   b) b, e, c, b, e
   c) a, a, b, e, d, e
   d) b, c, e, d, a, a, b
   e) b, c, c, b, e, d, e, d
   f) $a, a, b, b, c, c, b, e, d$

17. Find all circuits of length three in the directed graph in Exercise 16.

18. Determine whether there is a path in the directed graph in Exercise 16 beginning at the first vertex given and ending at the second vertex given.
   a) a, b
   b) b, a
   c) b, b
   d) a, e
   e) b, d
   f) c, d
   g) d, d
   h) e, a
   i) e, c

19. Let R be the relation on the set $\{1, 2, 3, 4, 5\}$ containing the ordered pairs $(1, 3)$, $(2, 4)$, $(3, 1)$, $(3, 5)$, $(4, 3)$, $(5, 1)$, $(5, 2)$, and $(5, 4)$. Find
   a) R^2.
   b) R^3.
   c) R^4.
   d) R^5.
   e) R^6.
   f) R^*.

20. Let R be the relation that contains the pair (a, b) if a and b are cities such that there is a direct non-stop airline flight from a to b. When is (a, b) in
   a) R^2?
   b) R^3?
   c) R^*?

21. Let R be the relation on the set of all students containing the ordered pair (a, b) if a and b are in at least one common class and $a \neq b$. When is (a, b) in
   a) R^2?
   b) R^3?
   c) R^*?

22. Suppose that the relation R is reflexive. Show that R^* is reflexive.

23. Suppose that the relation R is symmetric. Show that R^* is symmetric.

24. Suppose that the relation R is irreflexive. Is the relation R^2 necessarily irreflexive?

25. Use Algorithm 1 to find the transitive closures of the following relations on $\{1, 2, 3, 4\}$.
   a) $\{(1, 2), (2, 1), (2, 3), (3, 4), (4, 1)\}$
   b) $\{(2, 1), (2, 3), (3, 1), (3, 4), (4, 1), (4, 3)\}$
   c) $\{(1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4)\}$
   d) $\{(1, 1), (1, 4), (2, 1), (2, 3), (3, 1), (3, 2), (3, 4), (4, 2)\}$

26. Use Algorithm 1 to find the transitive closures of the following relations on $\{a, b, c, d, e\}$.
   a) $\{(a, c), (b, d), (c, a), (d, b), (e, d)\}$
   b) $\{(b, c), (b, e), (c, e), (d, a), (e, b), (e, c)\}$
   c) $\{(a, b), (a, c), (a, e), (b, a), (b, c), (c, a), (c, b), (d, a), (e, d)\}$
   d) $\{(a, e), (b, a), (b, d), (c, d), (d, a), (d, c), (e, a), (e, b), (e, c), (e, e)\}$

27. Use Warshall's algorithm to find the transitive closures of the relations in Exercise 25.

28. Use Warshall's algorithm to find the transitive closures of the relations in Exercise 26.

29. Find the smallest relation containing the relation $\{(1, 2), (1, 4), (3, 3), (4, 1)\}$ that is
   a) reflexive and transitive.
   b) symmetric and transitive.
   c) reflexive, symmetric, and transitive.

30. Finish the proof of the case when $a \neq b$ in Lemma 1.

31. Algorithms have been devised that use $O(n^{2.8})$ bit operations to compute the Boolean product of two $n \times n$ zero–one matrices. Assuming that these algorithms can be used, give big-O estimates for the number of bit operations using Algorithm 1 and using Warshall's algorithm to find the transitive closure of a relation on a set with n elements.

*32. Devise an algorithm using the concept of interior vertices in a path to find the length of the shortest path between two vertices in a directed graph, if such a path exists.

33. Adapt Algorithm 1 to find the reflexive closure of the transitive closure of a relation on a set with n elements.

34. Adapt Warshall's algorithm to find the reflexive closure of the transitive closure of a relation on a set with n elements.

35. Show that the closure with respect to the property **P** of the relation $R = \{(0, 0), (0, 1), (1, 1), (2, 2)\}$ on the set $\{0, 1, 2\}$ does not exist if **P** is the property
   a) "is not reflexive."
   b) "has an odd number of elements."

6.5

Equivalence Relations

INTRODUCTION

Students at a college register for classes the day before the start of a semester. Those with last names beginning with a letter from A to G, from H to N, and from O to Z may register at any time in the periods 8 A.M. to 11 A.M., 11 A.M. to 2 P.M., and 2 P.M. to 5 P.M., respectively. Let R be the relation containing (x, y) if and only if x and y are students with last names beginning with letters in the same block. Consequently, x and y can register at the same time if and only if (x, y) belongs to R. It is easy to see that R is reflexive, symmetric, and transitive. Furthermore, R divides the set of students into three classes, depending on the first letters of their last names. To know when a student can register we are concerned only with which of the three classes the student is in, and we do not care about the identity of the student.

The integers a and b are related by the "congruence modulo 4" relation when 4 divides $a - b$. We will show later that this relation is reflexive, symmetric, and transitive. It is not hard to see that a is related to b if and only if a and b have the same remainder when divided by 4. It follows that this relation splits the set of integers into four different classes. When we care only what remainder an integer leaves when it is divided by 4, we need only to know which class it is in, not its particular value.

These two relations, R and congruence modulo 4, are examples of equivalence relations, namely, relations that are reflexive, symmetric, and transitive. In this section we will show that such relations split sets into disjoint classes of equivalent elements. Equivalence relations arise whenever we care only whether an element of a set is in a certain class of elements, instead of caring about its particular identity.

EQUIVALENCE RELATIONS

In this section we will study relations with a particular combination of properties that allows them to be used to relate objects that are similar in some way.

DEFINITION 1. A relation on a set A is called an *equivalence relation* if it is reflexive, symmetric, and transitive.

Two elements that are related by an equivalence relation are called **equivalent.** (This definition makes sense since an equivalence relation is symmetric.) Since an equivalence relation is reflexive, in an equivalence relation every element is equivalent to itself. Furthermore, since an equivalence relation is transitive, if a and b are equivalent and b and c are equivalent, it follows that a and c are equivalent.

The following examples illustrate the notion of an equivalence relation.

EXAMPLE 1 Suppose that R is the relation on the set of strings of English letters such that $a\ R\ b$ if and only if $l(a) = l(b)$, where $l(x)$ is the length of the string x. Is R an equivalence relation?

Solution: Since $l(a) = l(a)$, it follows that $a\ R\ a$ whenever a is a string, so that R is reflexive. Next, suppose that $a\ R\ b$, so that $l(a) = l(b)$. Then $b\ R\ a$, since $l(b) = l(a)$. Hence, R is symmetric. Finally, suppose that $a\ R\ b$ and $b\ R\ c$. Then $l(a) = l(b)$ and $l(b) = l(c)$. Hence, $l(a) = l(c)$, so that $a\ R\ c$. Consequently, R is transitive. Since R is reflexive, symmetric, and transitive, it is an equivalence relation. ■

EXAMPLE 2 Let R be the relation on the set of integers such that $a\ R\ b$ if and only if $a = b$ or $a = -b$. In Section 6.1 we showed that R is reflexive, symmetric, and transitive. It follows that R is an equivalence relation. ■

EXAMPLE 3 Let R be the relation on the set of real numbers such that $a\ R\ b$ if and only if $a - b$ is an integer. Is R an equivalence relation?

Solution: Since $a - a = 0$ is an integer for all real numbers a, $a\ R\ a$ for all real numbers a. Hence, R is reflexive. Now suppose that $a\ R\ b$. Then $a - b$ is an integer, so that $b - a$ is also an integer. Hence, $b\ R\ a$. It follows that R is symmetric. If $a\ R\ b$ and $b\ R\ c$, then $a - b$ and $b - c$ are integers. Therefore, $a - c = (a - b) + (b - c)$ is also an integer. Hence, $a\ R\ c$. Thus, R is transitive. Consequently, R is an equivalence relation. ■

One of the most widely used equivalence relations is congruence modulo m, where m is a positive integer greater than 1.

EXAMPLE 4 Congruence Modulo m Let m be a positive integer greater than 1. Show that the relation

$$R = \{(a, b) \mid a \equiv b \pmod{m}\}$$

is an equivalence relation on the set of integers.

Solution: Recall from Section 2.3 that $a \equiv b \pmod{m}$ if and only if m divides $a - b$. Note that $a - a = 0$ is divisible by m, since $0 = 0 \cdot m$. Hence, $a \equiv a \pmod{m}$, so that congruence modulo m is reflexive. Now suppose that $a \equiv b \pmod{m}$. Then $a - b$ is divisible by m, so that $a - b = km$, where k is an integer. It follows that $b - a = (-k)m$, so that $b \equiv a \pmod{m}$. Hence, congruence modulo m is symmetric. Next, suppose that $a \equiv b \pmod{m}$ and $b \equiv c \pmod{m}$. Then m divides both $a - b$ and $b - c$. Therefore, there are integers k and l with $a - b = km$ and $b - c = lm$. Adding these two equations shows that $a - c = (a - b) + (b - c) = km + lm = (k + l)m$. Thus, $a \equiv c \pmod{m}$. Therefore, congruence modulo m is transitive. It follows that congruence modulo m is an equivalence relation. ■

EQUIVALENCE CLASSES

Let A be the set of all students in your school who graduated from high school. Consider the relation R on A that consists of all pairs (x, y) where x and y graduated from the same high school. Given a student x, we can form the set of all students equivalent to x with respect to R. This set consists of all students who graduated from the same high school as x did. This subset of A is called an equivalence class of the relation.

DEFINITION 2. Let R be an equivalence relation on a set A. The set of all elements that are related to an element a of A is called the *equivalence class* of a. The equivalence class of a with respect to R is denoted by $[a]_R$. When only one relation is under consideration, we will delete the subscript R and write $[a]$ for this equivalence class.

In other words, if R is an equivalence relation on a set A, the equivalence class of the element a is

$$[a]_R = \{s \mid (a, s) \in R\}.$$

If $b \in [a]_R$, b is called a **representative** of this equivalence class.

EXAMPLE 5 What is the equivalence class of an integer for the equivalence relation of Example 2?

Solution: Since an integer is equivalent to itself and its negative in this equivalence relation, it follows that $[a] = \{-a, a\}$. This set contains two distinct integers unless $a = 0$. For instance, $[7] = \{-7, 7\}$, $[-5] = \{-5, 5\}$, and $[0] = \{0\}$. ■

EXAMPLE 6 What are the equivalence classes of 0 and 1 for congruence modulo 4?

Solution: The equivalence class of 0 contains all integers a such that $a \equiv 0 \pmod 4$. The integers in this class are those divisible by 4. Hence, the equivalence class of 0 for this relation is

$$[0] = \{\ldots, -8, -4, 0, 4, 8, \ldots\}.$$

The equivalence class of 1 contains all the integers a such that $a \equiv 1 \pmod 4$. The integers in this class are those that have a remainder of 1 when divided by 4. Hence, the equivalence class of 1 for this relation is

$$[1] = \{\ldots, -7, -3, 1, 5, 9, \ldots\}.$$ ■

In Example 6 the equivalence classes of 0 and 1 with respect to congruence modulo 4 were found. Example 4 can easily be generalized, replacing 4 with any positive integer m. The equivalence classes of the relation congruence modulo m are called the **congruence classes modulo** m. The congruence class of an integer a modulo m is denoted by $[a]_m$ so that $[a]_m = \{\ldots, a-2m, a-m, a, a+m, a+2m, \ldots\}$. For instance, from Example 6 it follows that $[0]_4 = \{\ldots, -8, -4, 0, 4, 8, \ldots\}$ and $[1]_4 = \{-7, -3, 1, 5, 9, \ldots\}$.

EQUIVALENCE CLASSES AND PARTITIONS

Let A be the set of students at your school who are majoring in exactly one subject, and let R be the relation on A consisting of pairs (x, y) where x and y are students with the same major. Then R is an equivalence relation, as the reader should verify. We can see that R splits all students in A into a collection of disjoint subsets, where each subset contains students with a specified major. For instance, one subset contains all students majoring (just) in computer science, and a second subset contains all students majoring in history. Furthermore, these subsets are equivalence classes of R. This example illustrates how the equivalence classes of an equivalence relation partition a set into disjoint, nonempty subsets. We will make these notions more precise in the following discussion.

Let R be a relation on the set A. The following theorem shows that the equivalence classes of two elements of A are either identical or disjoint.

THEOREM 1 **Let R be an equivalence relation on a set A. The following statements are equivalent:**

(i) $a\ R\ b$
(ii) $[a] = [b]$
(iii) $[a] \cap [b] \neq \emptyset$

Proof: We first show that (i) implies (ii). Assume that $a\ R\ b$. We will prove that $[a] = [b]$ by showing $[a] \subseteq [b]$ and $[b] \subseteq [a]$. Suppose $c \in [a]$. Then $a\ R\ c$. Since $a\ R\ b$ and R is symmetric, we know that $b\ R\ a$. Furthermore, since R is transitive and $b\ R\ a$ and $a\ R\ c$, it follows that $b\ R\ c$. Hence, $c \in [b]$. This shows that $[a] \subseteq [b]$. The proof that $[b] \subseteq [a]$ is similar; it is left as an exercise for the reader.

Second, we will show that (ii) implies (iii). Assume that $[a] = [b]$. It follows that $[a] \cap [b] \neq \emptyset$ since $[a]$ is nonempty (since $a \in [a]$ because R is reflexive).

Next, we will show that (iii) implies (i). Suppose that $[a] \cap [b] \neq \emptyset$. Then there is an element c with $c \in [a]$ and $c \in [b]$. In other words, $a\ R\ c$ and $b\ R\ c$. By the symmetric property, $c\ R\ b$. Then by transitivity, since $a\ R\ c$ and $c\ R\ b$, we have $a\ R\ b$.

Since (i) implies (ii), (ii) implies (iii), and (iii) implies (i), the three statements, (i), (ii), and (iii), are equivalent. ☐

We are now in a position to show how an equivalence relation *partitions* a set. Let R be an equivalence relation on a set A. The union of the equivalence classes of R is all of A, since an element a of A is in its own equivalence class, namely, $[a]_R$. In other words,

$$\bigcup_{a \in A} [a]_R = A.$$

In addition, from Theorem 1, it follows that these equivalence classes are either equal or disjoint, so

$$[a]_R \cap [b]_R = \emptyset$$

when $[a]_R \neq [b]_R$.

These two observations show that the equivalence classes form a partition of A, since they split A into disjoint subsets. More precisely, a **partition** of a set S is a collection of disjoint nonempty subsets of S that have S as their union. In other words, the collection of subsets A_i, $i \in I$ (where I is an index set) forms a partition of S if and only if

$$A_i \neq \emptyset \text{ for } i \in I,$$

$$A_i \cap A_j = \emptyset, \text{ when } i \neq j,$$

and

$$\bigcup_{i \in I} A_i = S.$$

(Here the notation $\bigcup_{i \in I} A_i$ represents the union of the sets A_i for all $i \in I$.) Figure 1 illustrates the concept of a partition of a set.

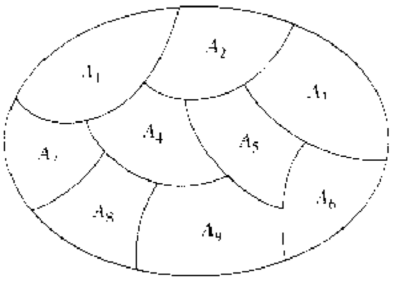


FIGURE 1 **A Partition of a Set.**

EXAMPLE 7 Suppose that $S = \{1, 2, 3, 4, 5, 6\}$. The collection of sets $A_1 = \{1, 2, 3\}$, $A_2 = \{4, 5\}$, and $A_3 = \{6\}$ forms a partition of S, since these sets are disjoint and their union is S. ■

We have seen that the equivalence classes of an equivalence relation on a set form a partition of the set. The subsets in this partition are the equivalence classes. Conversely, every partition of a set can be used to form an equivalence relation. Two elements are equivalent with respect to this relation if and only if they are in the same subset of the partition.

To see this, assume that $\{A_i \mid i \in I\}$ is a partition on S. Let R be the relation on S consisting of the pair (x, y) where x and y belong to the same subset A_i in the partition. To show that R is an equivalence relation we must show that R is reflexive, symmetric, and transitive.

We see that $(a, a) \in R$ for every $a \in S$, since a is in the same subset as itself. Hence, R is reflexive. If $(a, b) \in R$, then b and a are in the same subset of the partition, so that $(b, a) \in R$ as well. Hence, R is symmetric. If $(a, b) \in R$ and $(b, c) \in R$, then a and b are in the same subset in the partition, X, and b and c are in the same subset of the partition, Y. Since the subsets of the partition are disjoint, and b belongs to X and Y, it follows that $X = Y$. Consequently, a and c belong to the same subset of the partition, so that $(a, c) \in R$. Thus, R is transitive.

It follows that R is an equivalence relation. The equivalence classes of R consist of subsets of S containing related elements, and by the definition of R, these are the subsets of the partition. Theorem 2 summarizes the connections we have established between equivalence relations and partitions.

THEOREM 2 **Let R be an equivalence relation on a set S. Then the equivalence classes of R form a partition of S. Conversely, given a partition $\{A_i \mid i \in I\}$ of the set S, there is an equivalence relation R that has the sets A_i, $i \in I$, as its equivalence classes.**

The congruence classes modulo m provide a useful illustration of Theorem 2. There are m different congruence classes modulo m, corresponding to the m different remainders possible when an integer is divided by m. These m congruence classes are denoted by $[0]_m, [1]_m, \ldots, [m-1]_m$. They form a partition of the set of integers.

EXAMPLE 8 What are the sets in the partition of the integers arising from congruence modulo 4?

Solution: There are four congruence classes, corresponding to $[0]_4$, $[1]_4$, $[2]_4$, and $[3]_4$. They are the sets

$$[0]_4 = \{\ldots, -8, -4, 0, 4, 8, \ldots\}$$
$$[1]_4 = \{\ldots, -7, -3, 1, 5, 9, \ldots\}$$
$$[2]_4 = \{\ldots, -6, -2, 2, 6, 10, \ldots\}$$
$$[3]_4 = \{\ldots, -5, -1, 3, 7, 11, \ldots\}.$$

These congruence classes are disjoint, and every integer is in exactly one of them. In other words, as Theorem 2 says, these congruence classes form a partition. ■

Exercises

1. Which of the following relations on $\{0, 1, 2, 3\}$ are equivalence relations? Determine the properties of an equivalence relation that the others lack.

a) $\{(0, 0), (1, 1), (2, 2), (3, 3)\}$
b) $\{(0, 0), (0, 2), (2, 0), (2, 2), (2, 3), (3, 2), (3, 3)\}$
c) $\{(0, 0), (1, 1), (1, 2), (2, 1), (2, 2), (3, 3)\}$
d) $\{(0, 0), (1, 1), (1, 3), (2, 2), (2, 3), (3, 1), (3, 2), (3, 3)\}$
e) $\{(0, 0), (0, 1), (0, 2), (1, 0), (1, 1), (1, 2), (2, 0), (2, 2), (3, 3)\}$

2. Which of the following relations on the set of all people are equivalence relations? Determine the properties of an equivalence relation that the others lack.

a) $\{(a, b) \mid a$ and b are the same age$\}$
b) $\{(a, b) \mid a$ and b have the same parents$\}$
c) $\{(a, b) \mid a$ and b share a common parent$\}$
d) $\{(a, b) \mid a$ and b have met$\}$
e) $\{(a, b) \mid a$ and b speak a common language$\}$

3. Which of the following relations on the set of all functions from $\mathbf{Z}$ to $\mathbf{Z}$ are equivalence relations? Determine the properties of an equivalence relation that the others lack.

a) $\{(f, g) \mid f(1) = g(1)\}$
b) $\{(f, g) \mid f(0) = g(0)$ or $f(1) = g(1)\}$
c) $\{(f, g) \mid f(x) - g(x) = 1$ for all $x \in \mathbf{Z}\}$
d) $\{(f, g) \mid f(x) - g(x) = C$ for some $C \in \mathbf{Z}$ for all $x \in \mathbf{Z}\}$
e) $\{(f, g) \mid f(0) = g(1)$ and $f(1) = g(0)\}$

4. Define three equivalence relations on the set of students in your discrete mathematics class different from the relations discussed in the text. Determine the equivalence classes for these equivalence relations.

5. Suppose that A is a nonempty set, and f is a function that has A as its domain. Let R be the relation on A consisting of all ordered pairs (x, y) where $f(x) = f(y)$.

a) Show that R is an equivalence relation on A.
b) What are the equivalence classes of R?

6. Suppose that A is a nonempty set and R is an equivalence relation on A. Show that there is a function f with A as its domain such that $(x, y) \in R$ if and only if $f(x) = f(y)$.

7. Show that the relation R, consisting of all pairs (x, y) where x and y are bit strings of length three or more that agree in their first three bits, is an equivalence relation on the set of all bit strings of length three or more.

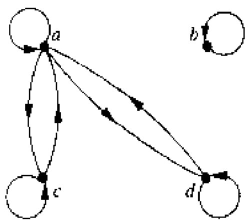
8. Show that the relation R, consisting of all pairs (x, y) where x and y are bit strings of length three or more that agree except perhaps in their first three bits, is an equivalence relation on the set of all bit strings.

9. Show that propositional equivalence is an equivalence relation on the set of all compound propositions.

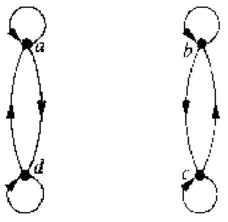
10. Let R be the relation on the set of ordered pairs of positive integers such that $((a, b), (c, d)) \in R$ if and only if $ad = bc$. Show that R is an equivalence relation.

In Exercises 11–13 determine whether the relation with the directed graphs shown is an equivalence relation.

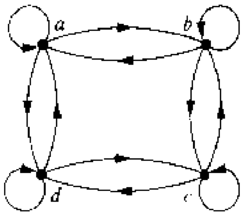
11.



12.



13.



14. Determine whether the relations represented by the following zero–one matrices are equivalence relations.

a) $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ **b)** $\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$ **c)** $\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

15. Show that the relation R on the set of all bit strings such that $s\,R\,t$ if and only if s and t contain the same number of 1s is an equivalence relation.

16. What are the equivalence classes of the equivalence relations in Exercise 1?

17. What are the equivalence classes of the equivalence relations in Exercise 2?

18. What are the equivalence classes of the equivalence relations in Exercise 3?

19. What is the equivalence class of the bit string 011 for the equivalence relation in Exercise 15?

20. What are the equivalence classes of the following bit strings for the equivalence relation in Exercise 7?

a) 010 **b)** 1011 **c)** 11111 **d)** 01010101

21. Describe the equivalence classes of the bit strings in Exercise 20 for the equivalence relation from Exercise 8.

22. What is the congruence class $[4]_m$ when m is

a) 2? **b)** 3? **c)** 6? **d)** 8?

23. Give a description of each of the congruence classes modulo 6.

24. **a)** What is the equivalence class of $(1, 2)$ with respect to the equivalence relation in Exercise 10?

b) Give an interpretation of the equivalence classes for the equivalence relation R in Exercise 10.

25. Which of the following collections of subsets are partitions of $\{1, 2, 3, 4, 5, 6\}$?

a) $\{1, 2\}, \{2, 3, 4\}, \{4, 5, 6\}$ **b)** $\{1\}, \{2, 3, 6\}, \{4\}, \{5\}$

c) $\{2, 4, 6\}, \{1, 3, 5\}$ **d)** $\{1, 4, 5\}, \{2, 6\}$

26. Which of the following collections of subsets are partitions of the set of integers?

a) the set of even integers and the set of odd integers

b) the set of positive integers and the set of negative integers

c) the set of integers divisible by 3, the set of integers leaving a remainder of 1 when divided by 3, and the set of integers leaving a remainder of 2 when divided by 3

d) the set of integers less than -100, the set of integers with absolute value not exceeding 100, and the set of integers greater than 100

e) the set of integers not divisible by 3, the set of even integers, and the set of integers that leave a remainder of 3 when divided by 6

A partition P_1 is called a **refinement** of the partition P_2 if every set in P_1 is a subset of one of the sets in P_2.

27. Show that the partition formed from the congruence classes modulo 6 is a refinement of the partition formed from the congruence classes modulo 3.

28. Suppose that R_1 and R_2 are equivalence relations on a set A. Let P_1 and P_2 be the partitions that correspond to R_1 and R_2, respectively. Show that $R_1 \subseteq R_2$ if and only if P_1 is a refinement of P_2.

29. Find the smallest equivalence relation on the set $\{a, b, c, d, e\}$ containing the relation $\{(a, b), (a, c), (d, e)\}$.

30. Suppose that R_1 and R_2 are equivalence relations on the set S. Determine whether each of the following combinations of R_1 and R_2 must be an equivalence relation.

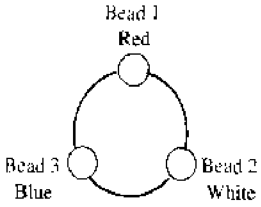
a) $R_1 \cup R_2$ **b)** $R_1 \cap R_2$ **c)** $R_1 \oplus R_2$

31. Consider the equivalence relation from Example 3, namely, $R = \{(x, y) \mid x - y \text{ is an integer}\}$.

a) What is the equivalence class of 1 for this equivalence relation?

b) What is the equivalence class of $1/2$ for this equivalence relation?

***32.** Each bead on a bracelet with three beads is either red, white, or blue, as illustrated in the example shown. Define the relation R between bracelets as follows: (B_1, B_2), where B_1 and B_2 are bracelets, belongs to R if and only if B_2 can be obtained from B_1 by rotating it or rotating it and then reflecting it.



a) Show that R is an equivalence relation.

b) What are the equivalence classes of R?

***33.** Let R be the relation on the set of all colorings of the 2×2 chessboard where each of the four squares is colored either red or blue so that (C_1, C_2), where C_1 and C_2 are 2×2 chessboards with each of their four squares colored blue or red, belongs to R if and only if C_2 can be obtained from C_1 either by rotating the chessboard or by rotating it and then reflecting it.

a) Show that R is an equivalence relation.

b) What are the equivalence classes of R?

34. **a)** Let R be the relation on the set of functions from $\mathbf{Z}^+$ to $\mathbf{Z}^+$ such that (f, g) belongs to R if and only if f is $\Theta(g)$ (see Section 1.8). Show that R is an equivalence relation.

b) Describe the equivalence class containing $f(n) = n^2$ for the equivalence relation of part (a).

35. Determine the number of different equivalence relations on a set with three elements by listing them.

36. Determine the number of different equivalence relations on a set with four elements by listing them.

***37.** Do we necessarily get an equivalence relation when we form the transitive closure of the symmetric closure of the reflexive closure of a relation?

***38.** Do we necessarily get an equivalence relation when we form the symmetric closure of the reflexive closure of the transitive closure of a relation?

39. Suppose we use Theorem 2 to form a partition P from an equivalence relation R. What is the equivalence relation R' that results if we use Theorem 2 again to form an equivalence relation from P?

40. Suppose we use Theorem 2 to form an equivalence relation R from a partition P. What is the partition P' that results if we use Theorem 2 again to form a partition from R?

41. Devise an algorithm to find the smallest equivalence relation containing a given relation.

***42.** Let $p(n)$ denote the number of different equivalence relations on a set with n elements (and by Theorem 2 the number of partitions of a set with n elements). Show that $p(n)$ satisfies the recurrence relation $p(n) = \sum_{j=0}^{n-1} C(n-1, j)p(n-j-1)$ and the initial condition $p(0) = 1$. [*Note:* The numbers $p(n)$ are called *Bell numbers* after the American mathematician E. T. Bell.]

43. Use Exercise 42 to find the number of different equivalence relations on a set with n elements where n is a positive integer not exceeding 10.

6.6

Partial Orderings

INTRODUCTION

We often use relations to order some or all of the elements of sets. For instance, we order words using the relation containing pairs of words (x, y) where x comes before y in the dictionary. We schedule projects using the relation consisting of pairs (x, y) where x and y are tasks in a project such that x must be completed before y begins. We order the set of integers using the relation containing the pairs (x, y) where x is less than y. When we add all of the pairs of the form (x, x) to these relations, we obtain a relation that is reflexive, antisymmetric, and transitive. These are properties that characterize relations used to order the elements of sets using their relative size.

DEFINITION 1. A relation R on a set S is called a *partial ordering* or *partial order* if it is reflexive, antisymmetric, and transitive. A set S together with a partial ordering R is called a *partially ordered set*, or *poset*, and is denoted by (S, R).

EXAMPLE 1 Show that the "greater than or equal" relation ($\geq$) is a partial ordering on the set of integers.

Solution: Since $a \geq a$ for every integer a, $\geq$ is reflexive. If $a \geq b$ and $b \geq a$, then $a = b$. Hence, $\geq$ is antisymmetric. Finally, $\geq$ is transitive since $a \geq b$ and $b \geq c$ imply that $a \geq c$. It follows that $\geq$ is a partial ordering on the set of integers and $(\mathbf{Z}, \geq)$ is a poset. ■

EXAMPLE 2 The divisibility relation $|$ is a partial ordering on the set of positive integers, since it is reflexive, antisymmetric, and transitive, as was shown in Section 6.1. We see that $(\mathbf{Z}^+, |)$ is a poset. ($\mathbf{Z}^+$ denotes the set of positive integers.) ■

EXAMPLE 3 Show that the inclusion relation $\subseteq$ is a partial ordering on the power set of a set S.

Solution: Since $A \subseteq A$ whenever A is a subset of S, $\subseteq$ is reflexive. It is antisymmetric since $A \subseteq B$ and $B \subseteq A$ imply that $A = B$. Finally, $\subseteq$ is transitive, since $A \subseteq B$ and $B \subseteq C$ imply that $A \subseteq C$. Hence, $\subseteq$ is a partial ordering on $P(S)$, and $(P(S), \subseteq)$ is a poset. ■

In a poset the notation $a \preccurlyeq b$ denotes that $(a, b) \in R$. This notation is used because the "less than or equal to" relation is a paradigm for a partial ordering. (Note that the symbol $\preccurlyeq$ is used to denote the relation in *any* poset, not just the "less than or equals" relation.) The notation $a \prec b$ denotes that $a \preccurlyeq b$, but $a \neq b$. Also, we say "a is less than b" or "b is greater than a" if $a \prec b$.

When a and b are elements of the poset $(S, \preccurlyeq)$, it is not necessary that either $a \preccurlyeq b$ or $b \preccurlyeq a$. For instance, in $(P(\mathbf{Z}), \subseteq)$, $\{1, 2\}$ is not related to $\{1, 3\}$, and vice versa, since neither set is contained within the other. Similarly, in $(\mathbf{Z}^+, |)$, 2 is not related to 3 and 3 is not related to 2, since $2 \nmid 3$ and $3 \nmid 2$. This leads to the following definition.

DEFINITION 2. **The elements a and b of a poset $(S, \preccurlyeq)$ are called *comparable* if either $a \preccurlyeq b$ or $b \preccurlyeq a$. When a and b are elements of S such that neither $a \preccurlyeq b$ nor $b \preccurlyeq a$, a and b are called *incomparable*.**

EXAMPLE 4 In the poset $(\mathbf{Z}^+, |)$, are the integers 3 and 9 comparable? Are 5 and 7 comparable?

Solution: The integers 3 and 9 are comparable, since $3 \mid 9$. The integers 5 and 7 are incomparable, because $5 \nmid 7$ and $7 \nmid 5$. ■

The adjective "partial" is used to describe partial orderings since pairs of elements may be incomparable. When every two elements in the set are comparable, the relation is called a **total ordering.**

DEFINITION 3. **If $(S, \preccurlyeq)$ is a poset and every two elements of S are comparable, S is called a *totally ordered* or *linearly ordered set*, and $\preccurlyeq$ is called a *total order* or a *linear order*. A totally ordered set is also called a *chain*.**

EXAMPLE 5 The poset $(\mathbf{Z}, \leq)$ is totally ordered, since $a \leq b$ or $b \leq a$ whenever a and b are integers. ■

EXAMPLE 6 The poset $(\mathbf{Z}^+, |)$ is not totally ordered since it contains elements that are incomparable, such as 5 and 7. ■

In Chapter 3 we noted that $(\mathbf{Z}^+, \leq)$ is well-ordered, where $\leq$ is the usual "less than or equals" relation. We now define well-ordered sets.

DEFINITION 4. **$(S, \preccurlyeq)$ is a *well-ordered set* if it is a poset such that $\preccurlyeq$ is a total ordering and such that every nonempty subset of S has a least element.**

EXAMPLE 7 The set of ordered pairs of positive integers, $\mathbf{Z}^+ \times \mathbf{Z}^+$, with $(a_1, a_2) \preccurlyeq (b_1, b_2)$ if $a_1 < b_1$, or if $a_1 = b_1$ and $a_2 \leq b_2$ (the lexicographic ordering), is a well-ordered set. The verification of this is left as an exercise at the end of this section. The set $\mathbf{Z}$, with the usual $\leq$ ordering, is not well-ordered since the set of negative integers, which is a subset of **Z**, has no least element. ■

LEXICOGRAPHIC ORDER

The words in a dictionary are listed in alphabetic, or lexicographic, order, which is based on the ordering of the letters in the alphabet. This is a special case of an ordering of strings on a set constructed from a partial ordering on the set. We will show how this construction works in any poset.

First, we will show how to construct a partial ordering on the Cartesian product of two posets, $(A_1, \preccurlyeq_1)$ and $(A_2, \preccurlyeq_2)$. The **lexicographic ordering** $\preccurlyeq$ on $A_1 \times A_2$ is defined by specifying that one pair is less than a second pair if the first entry of the first pair is less than (in A_1) the first entry of the second pair, or if the first entries are equal, but the second entry of this pair is less than (in A_2) the second entry of the second pair. In other words, (a_1, a_2) is less than (b_1, b_2), that is,

$$(a_1, a_2) \prec (b_1, b_2),$$

either if $a_1 \prec_1 b_1$ or if both $a_1 = b_1$ and $a_2 \prec_2 b_2$.

We obtain a partial ordering $\preccurlyeq$ by adding equality to the ordering $\prec$ on $A \times B$. The verification of this is left as an exercise.

EXAMPLE 8 Determine whether $(3, 5) \prec (4, 8)$, whether $(3, 8) \prec (4, 5)$, and whether $(4, 9) \prec (4, 11)$ in the poset $(\mathbf{Z} \times \mathbf{Z}, \preccurlyeq)$, where $\preccurlyeq$ is the lexicographic ordering constructed from the usual $\leq$ relation on **Z**.

Solution: Since $3 < 4$, it follows that $(3, 5) \prec (4, 8)$ and that $(3, 8) \prec (4, 5)$. We have $(4, 9) \prec (4, 11)$, since the first entries of $(4, 9)$ and $(4, 11)$ are the same, but $9 < 11$. ■

In Figure 1 the set of ordered pairs in $\mathbf{Z}^+ \times \mathbf{Z}^+$ that are less than $(3, 4)$ are highlighted.

A lexicographic ordering can be defined on the Cartesian product of n posets $(A_1, \preccurlyeq_1), (A_2, \preccurlyeq_2), \ldots, (A_n, \preccurlyeq_n)$. Define the partial ordering $\preccurlyeq$ on $A_1 \times A_2 \times \cdots \times A_n$ by

$$(a_1, a_2, \ldots, a_n) \prec (b_1, b_2, \ldots, b_n)$$

if $a_1 \prec_1 b_1$, or if there is an integer $i > 0$ such that $a_1 = b_1, \ldots, a_i = b_i$, and $a_{i+1} \prec_{i+1} b_{i+1}$. In other words, one n-tuple is less than a second n-tuple if the entry of the first n-tuple in the first position where the two n-tuples disagree is less than the entry in that position in the second n-tuple.

EXAMPLE 9 Note that $(1, 2, 3, 5) \prec (1, 2, 4, 3)$, since the entries in the first two positions of these 4-tuples agree, but in the third position the entry in the first 4-tuple, 3, is less than that in the second 4-tuple, 4. (Here the ordering on 4-tuples is the lexicographic ordering that comes from the usual "less than or equals" relation on the set of integers.) ■

We can now define lexicographic ordering of strings. Consider the strings $a_1a_2 \cdots a_m$ and $b_1b_2 \cdots b_n$ on a partially ordered set S. Suppose these strings are

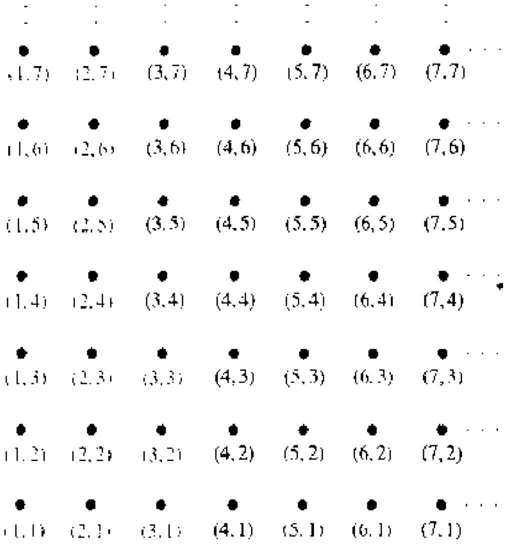


FIGURE 1 The Ordered Pairs Less Than (3, 4) in Lexicographic Order.

not equal. Let t be the minimum of m and n. The definition of lexicographic ordering is that the string $a_1a_2\cdots a_m$ is less than $b_1b_2\cdots b_n$ if and only if

$$(a_1, a_2, \ldots, a_t) \prec (b_1, b_2, \ldots, b_t), \text{ or}$$
$$(a_1, a_2, \ldots, a_t) = (b_1, b_2, \ldots, b_t) \text{ and } m < n,$$

where $\prec$ in this inequality represents the lexicographic ordering of S^t. In other words, to determine the ordering of two different strings, the longer string is truncated to the length of the shorter string, namely, to $t = \min(m, n)$ terms. Then the t-tuples made up of the first t terms of each string are compared using the lexicographic ordering on S^t. One string is less than another string if the t-tuple corresponding to the first string is less than the t-tuple of the second string, or if these two t-tuples are the same, but the second string is longer. The verification that this is a partial ordering is left as an exercise for the reader.

EXAMPLE 10 Consider the set of strings of lowercase English letters. Using the ordering of letters in the alphabet, a lexicographic ordering on the set of strings can be constructed. A string is less than a second string if the letter in the first string in the first position where the strings differ comes before the letter in the second string in this position, or if the first string and the second string agree in all positions, but the second string has more letters. This ordering is the same as that used in dictionaries. For example,

discreet $\prec$ *discrete*,

since these strings differ first in the seventh position, and $e \prec t$. Also,

discreet $\prec$ *discreetness*,

since the first eight letters agree, but the second string is longer. Furthermore,

discrete $\prec$ *discretion*,

since

discrete $\prec$ *discreti*. ■

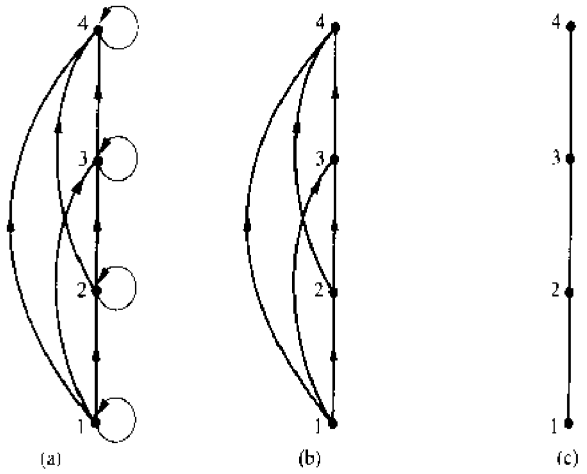


FIGURE 2 **Constructing the Hasse Diagram for ({1, 2, 3, 4}, ≤).**

HASSE DIAGRAMS

Many edges in the directed graph for a finite poset do not have to be shown since they must be present. For instance, consider the directed graph for the partial ordering $\{(a, b) \mid a \leq b\}$ on the set $\{1, 2, 3, 4\}$, shown in Figure 2(a). Since this relation is a partial ordering, it is reflexive, and its directed graph has loops at all vertices. Consequently, we do not have to show these loops since they must be present; in Figure 2(b) loops are not shown. Because a partial ordering is transitive, we do not have to show those edges that must be present because of transitivity. For example, in Figure 2(c) the edges (1, 3), (1, 4), and (2, 4) are not shown since they must be present. If we assume that all edges are pointed "upward" (as they are drawn in the figure), we do not have to show the directions of the edges; Figure 2(c) does not show directions.

In general, we can represent a partial ordering on a finite set using the following procedure. Start with the directed graph for this relation. Because a partial ordering is reflexive, a loop is present at every vertex. Remove these loops. Remove all edges that must be present because of the transitivity, since they must be present since a partial ordering is transitive. For instance, if (a, b) and (b, c) are in the partial ordering, remove the edge (a, c), since it must be present also. Furthermore, if (c, d) is also in the partial ordering, remove the edge (a, d), since is must be present also. Finally, arrange each

---

web **Helmut Hasse** (1898–1979). Helmut Hasse was born in Kassel. He served in the German navy after high school. He began his university studies at Göttingen University in 1918, moving in 1920 to Marburg University to study under the number theorist Kurt Hensel. During this time, Hasse made fundamental contributions to algebraic number theory. He became Hensel's successor at Marburg, later becoming director of the famous mathematical institute at Göttingen in 1934, and took a position at Hamburg University in 1950. Hasse served for 50 years as an editor of *Crelle's Journal*, a famous German mathematics periodical, taking over the job of chief editor in 1936 when the Nazis forced Hensel to resign. During World War II Hasse worked on applied mathematics research for the German navy. He was noted for the clarity and personal style of his lectures and was devoted both to number theory and to his students. (Hasse has been controversial for connections with the Nazi party. Investigations have shown he was a strong German nationalist but not an ardent Nazi.)

edge so that its initial vertex is below its terminal vertex (as it is drawn on paper). Remove all the arrows on the directed edges, since all edges point "upward" toward their terminal vertex. (The edges left correspond to pairs in the covering relation of the poset. See the preamble to Exercise 20.)

These steps are well-defined, and only a finite number of steps need to be carried out for a finite poset. When all the steps have been taken, the resulting diagram contains sufficient information to find the partial ordering. This diagram is called a **Hasse diagram**, named after the twentieth century German mathematician Helmut Hasse.

EXAMPLE 11 Draw the Hasse diagram representing the partial ordering $\{(a, b) \mid a \text{ divides } b\}$ on $\{1, 2, 3, 4, 6, 8, 12\}$.

Solution: Begin with the digraph for this partial order, as shown in Figure 3(a). Remove all loops, as shown in Figure 3(b). Then delete all the edges implied by the transitive property. These are $(1, 4)$, $(1, 6)$, $(1, 8)$, $(1, 12)$, $(2, 8)$, $(2, 12)$, and $(3, 12)$. Arrange all edges to point upward, and delete all arrows to obtain the Hasse diagram. The resulting Hasse diagram is shown in Figure 3(c). ■

EXAMPLE 12 Draw the Hasse diagram for the partial ordering $\{(A, B) \mid A \subseteq B\}$ on the power set $P(S)$ where $S = \{a, b, c\}$.

Solution: The Hasse diagram for this partial ordering is obtained from the associated digraph by deleting all the loops and all the edges that occur from transitivity, namely, $(\emptyset, \{a, b\})$, $(\emptyset, \{a, c\})$, $(\emptyset, \{b, c\})$, $(\emptyset, \{a, b, c\})$, $(\{a\}, \{a, b, c\})$, $(\{b\}, \{a, b, c\})$, and $(\{c\}, \{a, b, c\})$. Finally all edges point upward, and arrows are deleted. The resulting Hasse diagram is illustrated in Figure 4. ■

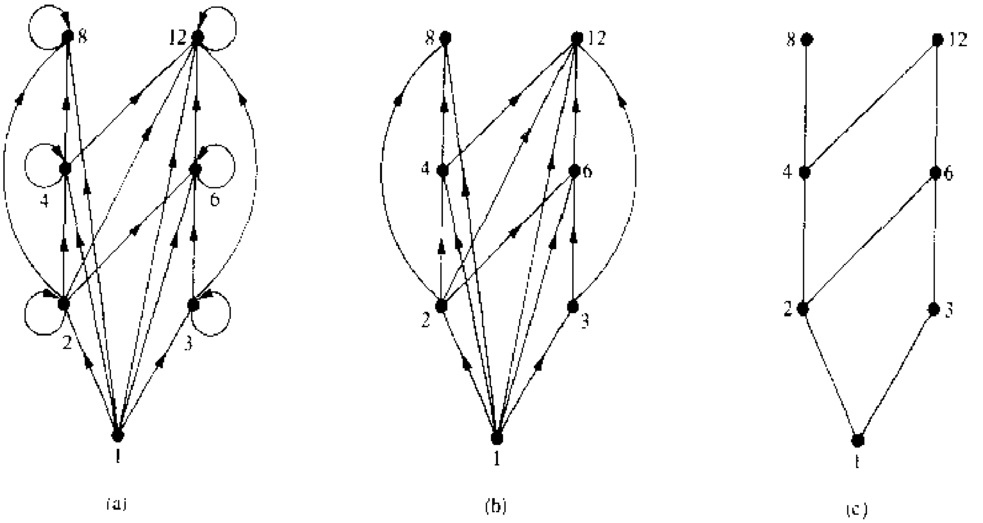


FIGURE 3 Constructing the Hasse Diagram of $(\{1, 2, 3, 4, 6, 8, 12\}, \mid)$.

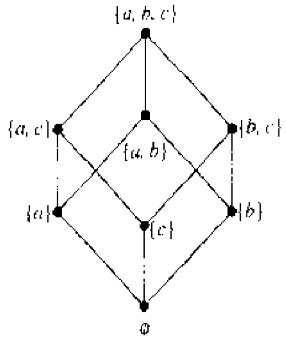


FIGURE 4 The Hasse Diagram of $(P(\{a, b, c\}), \subseteq)$.

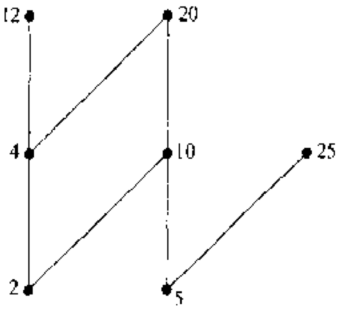


FIGURE 5 The Hasse Diagram of a Poset.

MAXIMAL AND MINIMAL ELEMENTS

Elements of posets that have certain extremal properties are important for many applications. An element of a poset is called maximal if it is not less than any element of the poset. That is, a is **maximal** in the poset $(S, \preccurlyeq)$ if there is no $b \in S$ such that $a \prec b$. Similarly, an element of a poset is called minimal if it is not greater than any element of the poset. That is, a is **minimal** if there is no element $b \in S$ such that $b \prec a$. Maximal and minimal elements are easy to spot using a Hasse diagram. They are the "top" and "bottom" elements in the diagram.

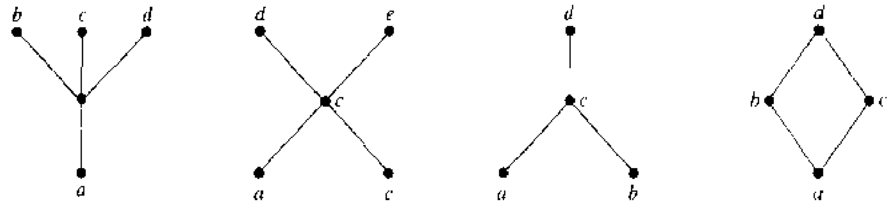
EXAMPLE 13 Which elements of the poset $(\{2, 4, 5, 10, 12, 20, 25\}, |)$ are maximal, and which are minimal?

Solution: The Hasse diagram in Figure 5 for this poset shows that the maximal elements are 12, 20, and 25, and the minimal elements are 2 and 5. As this example shows, a poset can have more than one maximal element and more than one minimal element. ■

Sometimes there is an element in a poset that is greater than every other element. Such an element is called the greatest element. That is, a is the **greatest element** of the poset $(S, \preccurlyeq)$ if $b \preccurlyeq a$ for all $b \in S$. The greatest element is unique when it exists [see Exercise 32(a) at the end of this section]. Likewise, an element is called the least element if it is less than all the other elements in the poset. That is, a is the **least element** of $(S, \preccurlyeq)$ if $a \preccurlyeq b$ for all $b \in S$. The least element is unique when it exists [see Exercise 32(b) at the end of the section].

EXAMPLE 14 Determine whether the posets represented by each of the Hasse diagrams in Figure 6 have a greatest element and a least element.

Solution: The least element of the poset with Hasse diagram (a) is a. This poset has no greatest element. The poset with Hasse diagram (b) has neither a least nor a greatest



(a) (b) (c) (d)

FIGURE 6 **Hasse Diagrams of Four Posets.**

element. The poset with Hasse diagram (c) has no least element. Its greatest element is d. The poset with Hasse diagram (d) has least element a and greatest element d. ■

EXAMPLE 15 Let S be a set. Determine whether there is a greatest element and a least element in the poset $(P(S), \subseteq)$.

Solution: The least element is the empty set, since $\emptyset \subseteq T$ for any subset T of S. The set S is the greatest element in this poset, since $T \subseteq S$ whenever T is a subset of S. ■

EXAMPLE 16 Is there a greatest element and a least element in the poset $(\mathbf{Z}^+, |)$?

Solution: The integer 1 is the least element since $1|n$ whenever n is a positive integer. Since there is no integer that is divisible by all positive integers, there is no greatest element. ■

Sometimes it is possible to find an element that is greater than all the elements in a subset A of a poset $(S, \preccurlyeq)$. If u is an element of S such that $a \preccurlyeq u$ for all elements $a \in A$, then u is called an **upper bound** of A. Likewise, there may be an element less than all the elements in A. If l is an element of S such that $l \preccurlyeq a$ for all elements $a \in A$, then l is called a **lower bound** of A.

EXAMPLE 17 Find the lower and upper bounds of the subsets $\{a, b, c\}$, $\{j, h\}$, and $\{a, c, d, f\}$ in the poset with the Hasse diagram shown in Figure 7.

Solution: The upper bounds of $\{a, b, c\}$ are e, f, j, and h, and its only lower bound is a. There are no upper bounds of $\{j, h\}$, and its lower bounds are a, b, c, d, e, and f. The upper bounds of $\{a, c, d, f\}$ are f, h, and j, and its lower bound is a. ■

The element x is called the **least upper bound** of the subset A if x is an upper bound that is less than every other upper bound of A. Since there is only one such element, if it exists, it makes sense to call this element *the* least upper bound [see Exercise 34(a) at the end of this section]. That is, x is the least upper bound of A if $a \preccurlyeq x$ whenever $a \in A$, and $x \preccurlyeq z$ whenever z is an upper bound of A. Similarly, the element y is

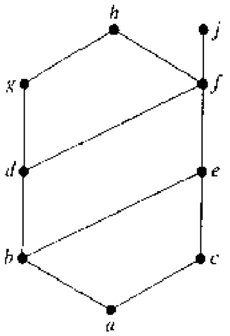


FIGURE 7 **The Hasse Diagram of a Poset.**

called the **greatest lower bound** of A if y is a lower bound of A and $z \preccurlyeq y$ whenever z is a lower bound of A. The greatest lower bound of A is unique if it exists [see Exercise 34(b) at the end of this section]. The greatest lower bound and least upper bound of a subset A are denoted by glb(A) and lub(A), respectively.

EXAMPLE 18 Find the greatest lower bound and the least upper bound of $\{b, d, g\}$, if they exist, in the poset shown in Figure 7.

Solution: The upper bounds of $\{b, d, g\}$ are g and h. Since $g \prec h$, g is the least upper bound. The lower bounds of $\{b, d, g\}$ are a and b. Since $a \prec b$, b is the greatest lower bound. ■

EXAMPLE 19 Find the greatest lower bound and the least upper bound of the sets $\{3, 9, 12\}$ and $\{1, 2, 4, 5, 10\}$ if they exist, in the poset $(\mathbf{Z}^+, |)$.

Solution: An integer is a lower bound of $\{3, 9, 12\}$ if 3, 9, and 12 are divisible by this integer. The only such integers are 1 and 3. Since $1 \mid 3$, 3 is the greatest lower bound of $\{3, 9, 12\}$. The only lower bound for the set $\{1, 2, 4, 5, 10\}$ with respect to $|$ is the element 1. Hence, 1 is the greatest lower bound for $\{1, 2, 4, 5, 10\}$.

An integer is an upper bound for $\{3, 9, 12\}$ if and only if it is divisible by 3, 9, and 12. The integers with this property are those divisible by the least common multiple of 3, 9, and 12, which is 36. Hence, 36 is the least upper bound of $\{3, 9, 12\}$. A positive integer is an upper bound for the set $\{1, 2, 4, 5, 10\}$ if and only if it is divisible by 1, 2, 4, 5, and 10. The integers with this property are those integers divisible by the least common multiple of these integers, which is 20. Hence, 20 is the least upper bound of $\{1, 2, 4, 5, 10\}$. ■

LATTICES

A partially ordered set in which every pair of elements has both a least upper bound and a greatest lower bound is called a **lattice.** Lattices have many special properties. Furthermore, lattices are used in many different applications such as models of information flow and play an important role in Boolean algebra.

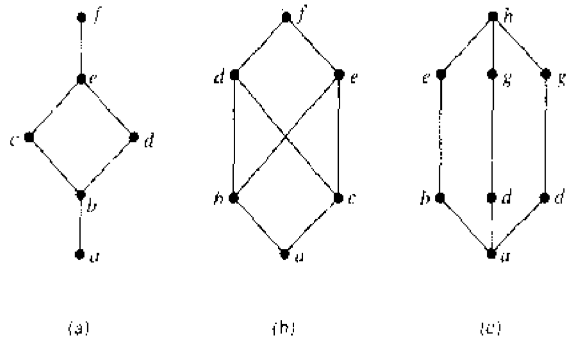


FIGURE 8 **Hasse Diagrams of Three Posets.**

EXAMPLE 20 Determine whether the posets represented by each of the Hasse diagrams in Figure 8 are lattices.

Solution: The posets represented by the Hasse diagrams in (a) and (c) are both lattices because in each poset every pair of elements has both a least upper bound and a greatest lower bound, as the reader should verify. On the other hand, the poset with the Hasse diagram shown in (b) is not a lattice, since the elements b and c have no least upper bound. To see this note that each of the elements d, e, and f is an upper bound, but none of these three elements precedes the other two with respect to the ordering of this poset. ■

EXAMPLE 21 Is the poset $(\mathbf{Z}^+, |)$ a lattice?

Solution: Let a and b be two positive integers. The least upper bound and greatest lower bound of these two integers are the least common multiple and the greatest common divisor of these integers, respectively, as the reader should verify. It follows that this poset is a lattice. ■

EXAMPLE 22 Determine whether each of the posets $(\{1, 2, 3, 4, 5\}, |)$ and $(\{1, 2, 4, 8, 16\}, |)$ is a lattice.

Solution: Since 2 and 3 have no upper bounds in $(\{1, 2, 3, 4, 5\}, |)$, they certainly do not have a least upper bound. Hence, the first poset is not a lattice.

Every two elements of the second poset have both a least upper bound and a greatest lower bound. The least upper bound of two elements in this poset is the larger of the elements and the greatest lower bound of two elements is the smaller of the elements, as the reader should verify. Hence this second poset is a lattice. ■

EXAMPLE 23 Determine whether $(P(S), \subseteq)$ is a lattice where S is a set.

Solution: Let A and B be two subsets of S. The least upper bound and the greatest lower bound of A and B are $A \cup B$ and $A \cap B$, respectively, as the reader can show. Hence $(P(S), \subseteq)$ is a lattice. ■

EXAMPLE 24 *The Lattice Model of Information Flow* In many settings the flow of information from one person or computer program to another is restricted via security clearances. We can use a lattice model to represent different information flow policies. For example, one common information flow policy is the *multilevel security policy* used in government and military systems. Each piece of information is assigned to a security class, and each security class is represented by a pair (A, C) where A is an *authority level* and C is a *category*. People and computer programs are then allowed access to information from a specific restricted set of security classes.

The typical authority levels used in the U.S. government are unclassified (0), confidential (1), secret (2), and top secret (3). Categories used in security classes are the subsets of a set of all *compartments* relevant to a particular area of interest. Each compartment represents a particular subject area. For example, if the set of compartments is {*spies, moles, double agents*}, then there are eight different categories, one for each of the eight subsets of the set of compartments, such as {*spies, moles*}.

We can order security classes by specifying that $(A_1, C_1) \preccurlyeq (A_2, C_2)$ if and only if $A_1 \leq A_2$ and $C_1 \subseteq C_2$. Information is permitted to flow from security class (A_1, C_1) into security class (A_2, C_2) if and only if $(A_1, C_1) \preccurlyeq (A_2, C_2)$. For example, information is permitted to flow from the security class (*secret*, {*spies, moles*}) into the security class (*top secret*, {*spies, moles, double agents*}), whereas information is not allowed to flow from the security class (*top secret*, {*spies, moles* }) into either of the security classes (*secret*, {*spies, moles, double agents*}) or (*top secret*,{*spies*}).

We leave it to the reader (see Exercise 40 at the end of this section) to show that the set of all security classes with the ordering defined in this example forms a lattice. ■

TOPOLOGICAL SORTING

Suppose that a project is made up of 20 different tasks. Some tasks can be completed only after others have been finished. How can an order be found for these tasks? To model this problem we set up a partial order on the set of tasks, so that $a \prec b$ if and only if a and b are tasks where b cannot be started until a has been completed. To produce a schedule for the project, we need to produce an order for all 20 tasks that is compatible with this partial order. We will show how this can be done.

We begin with a definition. A total ordering $\preccurlyeq$ is said to be **compatible** with the partial ordering R if $a \preccurlyeq b$ whenever $a \mathrel{R} b$. Constructing a compatible total ordering from a partial ordering is called **topological sorting**. We will need to use the following lemma.

LEMMA 1 **Every finite nonempty poset $(S, \preccurlyeq)$ has a minimal element.**

Proof: Choose an element a_0 of S. If a_0 is not minimal, then there is an element a_1 with $a_1 \prec a_0$. If a_1 is not minimal, there is an element a_2 with $a_2 \prec a_1$. Continue this process, so that if a_n is not minimal, there is an element a_{n+1} with $a_{n+1} \prec a_n$. Since there are only a finite number of elements in the poset, this process must end with a minimal element a_n. □

The topological sorting algorithm we will describe works for any finite nonempty poset. To define a total ordering on the poset $(A, \preccurlyeq)$, first choose a minimal element

a_1; such an element exists by Lemma 1. Next, note that $(A - \{a_1\}, \preccurlyeq)$ is also a poset, as the reader should verify. If it is nonempty, choose a minimal element a_2 of this poset. Then remove a_2 as well, and if there are additional elements left, choose a minimal element a_3 in $A - \{a_1, a_2\}$. Continue this process by choosing a_{k+1} to be a minimal element in $A - \{a_1, a_2, \ldots, a_k\}$, as long as elements remain.

Since A is a finite set, this process must terminate. The end product is a sequence of elements $a_1, a_2, \ldots, a_n$. The desired total ordering is defined by

$$a_1 \preccurlyeq a_2 \preccurlyeq \cdots \preccurlyeq a_n.$$

This total ordering is compatible with the original partial ordering. To see this, note that if $b \prec c$ in the original partial ordering, c is chosen as the minimal element at a phase of the algorithm where b has already been removed, for otherwise c would not be a minimal element. Pseudocode for this topological sorting algorithm is shown in Algorithm 1.

ALGORITHM 1 Topological Sorting.

```
procedure topological sort ($S$: finite poset)
$k := 1$
while $S \neq \emptyset$
begin
    $a_k :=$ a minimal element of $S$ {such an element exists by Lemma 1}
    $S := S - \{a_k\}$
    $k := k + 1$
end {$a_1, a_2, \ldots, a_n$ is a compatible total ordering of $S$}
```

EXAMPLE 25 Find a compatible total ordering for the poset $(\{1, 2, 4, 5, 12, 20\}, |)$.

Solution: The first step is to choose a minimal element. This must be 1, since it is the only minimal element. Next, select a minimal element of $(\{2, 4, 5, 12, 20\}, |)$. There are two minimal elements in this poset, namely, 2 and 5. We select 5. The remaining elements are $\{2, 4, 12, 20\}$. The only minimal element at this stage is 2. Next, 4 is chosen since it is the only minimal element of $(\{4, 12, 20\}, |)$. Since both 12 and 20 are minimal elements of $(\{12, 20\}, |)$, either can be chosen next. We select 20, which leaves 12 as the last element left. This produces the total ordering

$$1 \prec 5 \prec 2 \prec 4 \prec 20 \prec 12.$$

The steps used by this sorting algorithm are displayed in Figure 9. ■

Topological sorting has an application to the scheduling of projects.

EXAMPLE 26 A development project at a computer company requires the completion of seven tasks. Some of these tasks can be started only after other tasks are finished. A partial ordering on tasks is set up by considering task $X \prec$ task Y if task Y cannot be started until task X has been completed. The Hasse diagram for the seven tasks, with respect to this partial

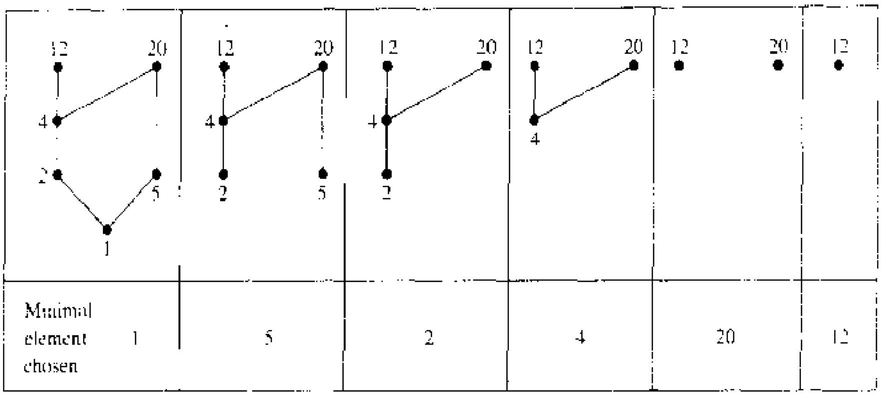

12
20
4
2
5
1
Minimal element chosen
1
5
2
4
20
12

FIGURE 9 A Topological Sort of ({1, 2, 4, 5, 12, 20}, |).

ordering, is shown in Figure 10. Find an order in which these tasks can be carried out to complete the project.

Solution: An ordering of the seven tasks can be obtained by performing a topological sort. The steps of a sort are illustrated in Figure 11. The result of this sort, $A \prec C \prec B \prec E \prec F \prec D \prec G$, gives one possible order for the tasks. ■

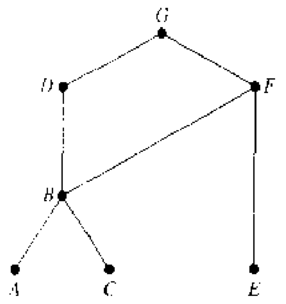


FIGURE 10 The Hasse Diagram for Seven Tasks.

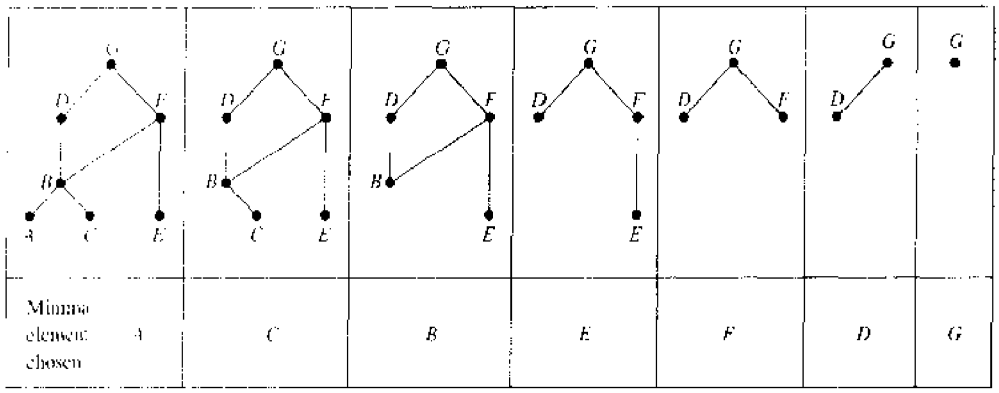

G
D
F
B
A
C
E
Minimal element chosen
A
C
B
E
F
D
G

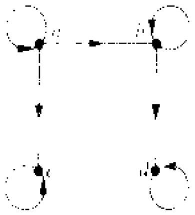
FIGURE 11 A Topological Sort of the Tasks.

Exercises

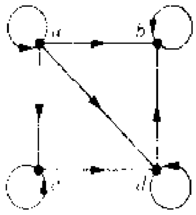
1. Which of the following are posets?
   a) $(\mathbf{Z}, =)$ **b)** $(\mathbf{Z}, \neq)$ **c)** $(\mathbf{Z}, \geq)$ **d)** $(\mathbf{Z}, \nmid)$
2. Determine whether the relations represented by the following zero-one matrices are partial orders.
   a) $\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ **b)** $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$
   c) $\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$

In Exercises 3–5 determine whether the relation with the directed graph shown is a partial order.

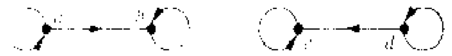
3.



4.



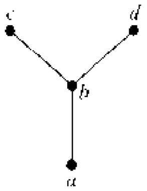
5.



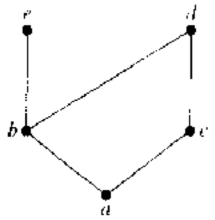
6. Let (S, R) be a poset. Show that (S, R^{-1}) is also a poset, where R^{-1} is the inverse of R. The poset (S, R^{-1}) is called the **dual** of (S, R).
7. Find the duals of the following posets.
   a) $(\{0, 1, 2\}, \leq)$
   b) $(\mathbf{Z}, \geq)$
   c) $(P(\mathbf{Z}), \supseteq)$
   d) $(\mathbf{Z}^+, |)$
8. Which of the following pairs of elements are comparable in the poset $(\mathbf{Z}^+, |)$?
   a) 5, 15 **b)** 6, 9 **c)** 8, 16 **d)** 7, 7
9. Find two incomparable elements in the following posets.
   a) $(P(\{0, 1, 2\}), \subseteq)$
   b) $(\{1, 2, 4, 6, 8\}, |)$
10. Let $S = \{1, 2, 3, 4\}$. With respect to the lexicographic order based on the usual "less than" relation,
    a) find all pairs in $S \times S$ less than $(2, 3)$.
    b) find all pairs in $S \times S$ greater than $(3, 1)$.
    c) draw the Hasse diagram of the poset $(S \times S, \preccurlyeq)$.
11. Find the lexicographic ordering of the following n-tuples.
    a) $(1, 1, 2), (1, 2, 1)$
    b) $(0, 1, 2, 3), (0, 1, 3, 2)$
    c) $(1, 0, 1, 0, 1), (0, 1, 1, 1, 0)$
12. Find the lexicographic ordering of the following strings of lowercase English letters:
    a) *quack, quick, quicksilver, quicksand, quacking*
    b) *open, opener, opera, operand, opened*
    c) *zoo, zero, zoom, zoology, zoological*
13. Find the lexicographic ordering of the bit strings 0, 01, 11, 001, 010, 011, 0001, and 0101 based on the ordering $0 < 1$.
14. Draw the Hasse diagram for the "greater than or equals" relation on $\{0, 1, 2, 3, 4, 5\}$.
15. Draw the Hasse diagram for divisibility on the set
    a) $\{1, 2, 3, 4, 5, 6, 7, 8\}$. **b)** $\{1, 2, 3, 5, 7, 11, 13\}$.
    c) $\{1, 2, 3, 6, 12, 24, 36, 48\}$. **d)** $\{1, 2, 4, 8, 16, 32, 64\}$.
16. Draw the Hasse diagram for inclusion on the set $P(S)$ where $S = \{a, b, c, d\}$.

In Exercises 17–19 list all ordered pairs in the partial ordering with the accompanying Hasse diagram.

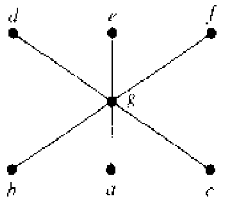
17.



18.

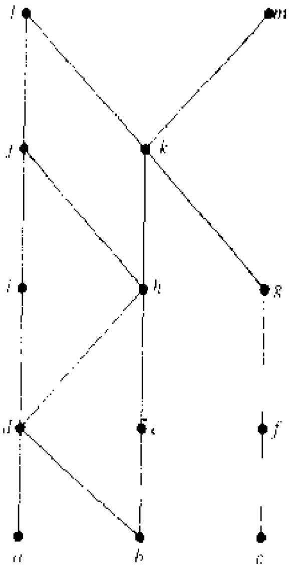


19.



Let $(S, \preccurlyeq)$ be a poset. We say that an element $y \in S$ **covers** an element $x \in S$ if $x \prec y$ and there is no element $z \in S$ such that $x \prec z \prec y$. The set of pairs (x, y) such that y covers x is called the **covering relation** of $(S, \preccurlyeq)$.

20. What is the covering relation of the partial ordering $\{(a, b) \mid a \text{ divides } b\}$ on $\{1, 2, 3, 4, 6, 12\}$?
21. What is the covering relation of the partial ordering $\{(A, B) \mid A \subseteq B\}$ on the power set of S where $S = \{a, b, c\}$?
22. Show that the pair (x, y) belongs to the covering relation of the finite poset $(S, \preccurlyeq)$ if and only if x is lower than y and there is an edge joining x and y in the Hasse diagram of this poset.
23. Show that a finite poset can be reconstructed from its covering relation.
24. Answer the following questions for the partial order represented by the following Hasse diagram.



a) Find the maximal elements.
b) Find the minimal elements.
c) Is there a greatest element?
d) Is there a least element?
e) Find all upper bounds of $\{a, b, c\}$.
f) Find the least upper bound of $\{a, b, c\}$, if it exists.
g) Find all lower bounds of $\{f, g, h\}$.
h) Find the greatest lower bound of $\{f, g, h\}$, if it exists.

25. Answer the following questions concerning the poset $(\{3, 5, 9, 15, 24, 45\}, |)$.
a) Find the maximal elements.
b) Find the minimal elements.
c) Is there a greatest element?
d) Is there a least element?
e) Find all upper bounds of $\{3, 5\}$.
f) Find the least upper bound of $\{3, 5\}$, if it exists.
g) Find all lower bounds of $\{15, 45\}$.
h) Find the greatest lower bound of $\{15, 45\}$, if it exists.

26. Answer the following questions concerning the poset $(\{2, 4, 6, 9, 12, 18, 27, 36, 48, 60, 72\}, |)$.
a) Find the maximal elements.
b) Find the minimal elements.
c) Is there a greatest element?
d) Is there a least element?
e) Find all upper bounds of $\{2, 9\}$.
f) Find the least upper bound of $\{2, 9\}$, if it exists.
g) Find all lower bounds of $\{60, 72\}$.
h) Find the greatest lower bound of $\{60, 72\}$, if it exists.

27. Answer the following questions concerning the poset $(\{\{1\}, \{2\}, \{4\}, \{1, 2\}, \{1, 4\}, \{2, 4\}, \{3, 4\}, \{1, 3, 4\}, \{2, 3, 4\}\}, \subseteq)$.
a) Find the maximal elements.
b) Find the minimal elements.
c) Is there a greatest element?
d) Is there a least element?
e) Find all upper bounds of $\{\{2\}, \{4\}\}$.
f) Find the least upper bound of $\{\{2\}, \{4\}\}$, if it exists.
g) Find all lower bounds of $\{\{1, 3, 4\}, \{2, 3, 4\}\}$.
h) Find the greatest lower bound of $\{\{1, 3, 4\}, \{2, 3, 4\}\}$, if it exists.

28. Give a poset that has
a) a minimal element but no maximal element.
b) a maximal element but no minimal element.
c) neither a maximal nor a minimal element.

29. Show that lexicographic order is a partial ordering on the Cartesian product of two posets.

30. Show that lexicographic order is a partial ordering on the set of strings from a poset.

31. Suppose that $(S, \preccurlyeq_1)$ and $(T, \preccurlyeq_2)$ are posets. Show that $(S \times T, \preccurlyeq)$ is a poset where $(s, t) \preccurlyeq (u, v)$ if and only if $s \preccurlyeq_1 u$ and $t \preccurlyeq_2 v$.

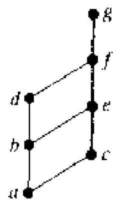
32. a) Show that there is exactly one greatest element of a poset, if such an element exists.
b) Show that there is exactly one least element of a poset, if such an element exists.

33. a) Show that there is exactly one maximal element in a poset with a greatest element.
b) Show that there is exactly one minimal element in a poset with a least element.

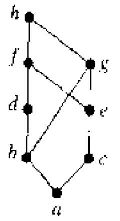
34. a) Show that the least upper bound of a set in a poset is unique if it exists.
b) Show that the greatest lower bound of a set in a poset is unique if it exists.

35. Determine whether the posets with the following Hasse diagrams are lattices.

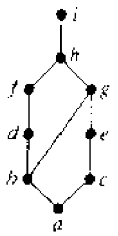
a)



b)



c)



36. Determine whether the following posets are lattices.
a) $(\{1, 3, 6, 9, 12\}, |)$
b) $(\{1, 5, 25, 125\}, |)$
c) $(\mathbf{Z}, \geq)$
d) $(P(S), \supseteq)$, where $P(S)$ is the power set of a set S

37. Show that every nonempty subset of a lattice has a least upper bound and a greatest lower bound.

38. Show that if the poset (S, R) is a lattice then the dual poset (S, R^{-1}) is also a lattice.

39. In a company, the lattice model of information flow is used to control sensitive information with security classes represented by ordered pairs (A, C). Here A is an authority level which may be nonproprietary (0), proprietary (1), restricted (2), or registered (3).

A category C is a subset of the set of all projects $\{Cheetah, Impala, Puma\}$. (Names of animals are often used as code names for projects in companies.)

a) Is information permitted to flow from $(Proprietary, \{Cheetah, Puma\})$ into $(Restricted, \{Puma\})$?
b) Is information permitted to flow from $(Restricted, \{Cheetah\})$ into $(Registered, \{Cheetah, Impala\})$?
c) Into which classes is information from $(Proprietary, \{Cheetah, Puma\})$ permitted to flow?
d) From which classes is information permitted to flow into the security class $(Restricted, \{Impala, Puma\})$?

40. Show that the set S of security classes (A, C) is a lattice, where A is a positive integer representing an authority class and C is a subset of a finite set of compartments, with $(A_1, C_1) \preccurlyeq (A_2, C_2)$ if and only if $A_1 \leq A_2$ and $C_1 \subseteq C_2$. [*Hint:* First show that $(S, \preccurlyeq)$ is a poset and then show that the least upper bound and greatest lower bound of (A_1, C_1) and (A_2, C_2) are $(\max(A_1, A_2), C_1 \cup C_2)$ and $(\min(A_1, A_2), C_1 \cap C_2)$, respectively.]

***41.** Show that the set of all partitions of a set S with the relation $P_1 \preccurlyeq P_2$ if the partition P_1 is a refinement of the partition P_2 is a lattice. (See the preamble to Exercise 27 of Section 6.5.)

42. Show that every totally ordered set is a lattice.

43. Show that every finite lattice has a least element and a greatest element.

44. Give an example of an infinite lattice with

a) neither a least nor a greatest element.
b) a least but not a greatest element.
c) a greatest but not a least element.
d) both a least and a greatest element.

45. Verify that $(\mathbf{Z}^+ \times \mathbf{Z}^+, \preccurlyeq)$ is a well-ordered set, where $\preccurlyeq$ is lexicographic order, as claimed in Example 7.

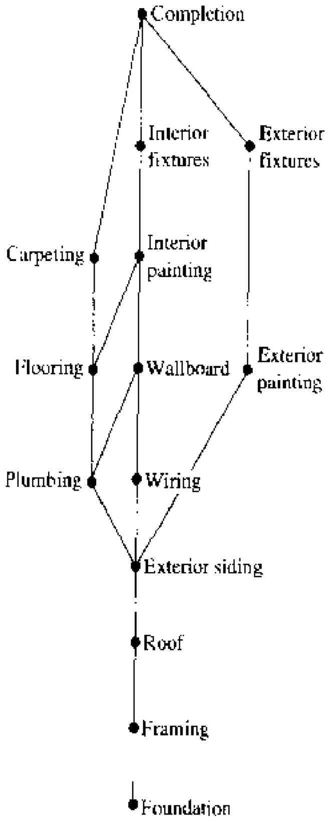
46. Show that a finite nonempty poset has a maximal element.

47. Find a compatible total order for the poset with the Hasse diagram shown in Exercise 24.

48. Find a compatible total order for the divisibility relation on the set $\{1, 2, 3, 6, 8, 12, 24, 36\}$.

49. Find an order different from that constructed in Example 26 for completing the tasks in the development project.

50. Schedule the tasks needed to build a house, by specifying their order, if the Hasse diagram representing these tasks is as shown in the following figure.



Key Terms and Results

TERMS

binary relation from A to B: a subset of $A \times B$

relation on A: a binary relation from A to itself (i.e., a subset of $A \times A$)

$S \circ R$: composite of R and S

R^{-1}: inverse relation of R

R^n: nth power of R

reflexive: a relation R on A is reflexive if $(a, a) \in R$ for all $a \in A$

symmetric: a relation R on A is symmetric if $(b, a) \in R$ whenever $(a, b) \in R$

antisymmetric: a relation R on A is antisymmetric if $a = b$ whenever $(a, b) \in R$ and $(b, a) \in R$

transitive: a relation R on A is transitive if $(a, b) \in R$ and $(b, c) \in R$ implies that $(a, c) \in R$

n-ary relation on $A_1, A_2, \ldots, A_n$: a subset of $A_1 \times A_2 \times \cdots \times A_n$

relational data model: a model for representing databases using n-ary relations

primary key: a domain of an n-ary relation such that an n-tuple is uniquely determined by its value for this domain

composite key: the Cartesian product of domains of an n-ary relation such that an n-tuple is uniquely determined by its values in these domains

projection: a function that produces relations of smaller degree from an n-ary relation by deleting fields

join: a function that combines n-ary relations that agree on certain fields

directed graph or digraph: a set of elements called vertices and ordered pairs of these elements, called edges

loop: an edge of the form (a, a)

closure of a relation R with respect to a property P: the relation S (if it exists) that contains R, has property **P**, and is contained within any relation that contains R and has property **P**

path in a digraph: a sequence of edges $(a, x_1), (x_1, x_2), \ldots, (x_{n-2}, x_{n-1}), (x_{n-1}, b)$ such that the terminal vertex of each edge is the initial vertex of the succeeding edge in the sequence

circuit (or cycle) in a digraph: a path that begins and ends at the same vertex

R^* (connectivity relation): the relation consisting of those ordered pairs (a, b) such that there is a path from a to b

equivalence relation: a reflexive, symmetric, and transitive relation

equivalent: if R is an equivalence relation, a is equivalent to b if $a\,R\,b$

$[a]_R$ (equivalence class of a with respect to R): the set of all elements of A that are equivalent to a

$[a]_m$ (congruence class modulo m): the set of integers congruent to a modulo m

partition of a set S: a collection of pairwise disjoint nonempty subsets that have S as their union

partial ordering: a relation that is reflexive, antisymmetric, and transitive

poset (S, R): a set S and a partial ordering R on this set

comparable: the elements a and b in the poset $(A, \preccurlyeq)$ are comparable if $a \preccurlyeq b$ or $b \preccurlyeq a$

incomparable: elements in a poset that are not comparable

total (or linear) ordering: a partial ordering for which every pair of elements are comparable

totally (or linearly) ordered set: a poset with a total (or linear) ordering

lexicographic order: a partial ordering of Cartesian products or strings (see pages 417–418)

Hasse diagram: a graphical representation of a poset where loops and all edges resulting from the transitive property are not shown, and the direction of the edges is indicated by the position of the vertices

maximal element: an element of a poset that is not less than any other element of the poset

minimal element: an element of a poset that is not greater than any other element of the poset

least element: an element of a poset less than or equal to all other elements in this set

greatest element: an element of a poset greater than or equal to all other elements in this set

upper bound of a set: an element in a poset greater than all other elements in the set

lower bound of a set: an element in a poset less than all other elements in the set

least upper bound of a set: an upper bound of the set that is less than all other upper bounds

greatest lower bound of a set: a lower bound of the set that is greater than all other lower bounds

lattice: a partially ordered set in which every two elements have a greatest lower bound and a least upper bound

well-ordered set: a poset $(S, \preccurlyeq)$ where $\preccurlyeq$ is a total order and every nonempty subset of S has a least element

compatible total ordering for a partial ordering: a total ordering that contains the given partial ordering

topological sort: the construction of a total ordering compatible with a given partial ordering

RESULTS

The reflexive closure of a relation R on the set A equals $R \cup \Delta$, where $\Delta = \{(a, a) \mid a \in A\}$.

The symmetric closure of a relation R on the set A equals $R \cup R^{-1}$, where $R^{-1} = \{(b, a) \mid (a, b) \in R\}$.

The transitive closure of a relation equals the connectivity relation formed from this relation.

Warshall's algorithm for finding the transitive closure of a relation (see pages 403–406).

Let R be an equivalence relation. Then the following three statements are equivalent: (1) $a\,R\,b$; (2) $[a]_R \cap [b]_R \neq \emptyset$; (3) $[a]_R = [b]_R$.

The equivalence classes of an equivalence relation on a set A form a partition of A. Conversely, an equivalence relation can be constructed from any partition so that the equivalence classes are the subsets in the partition.

The topological sorting algorithm (see pages 425–427).

Review Questions

1. a) What is a relation on a set?
   b) How many relations are there on a set with n elements?
2. a) What is a reflexive relation?
   b) What is a symmetric relation?
   c) What is an antisymmetric relation?
   d) What is a transitive relation?
3. Give an example of a relation on the set $\{1, 2, 3, 4\}$ that is
   a) reflexive, symmetric, and not transitive.

b) not reflexive, symmetric, and transitive.
c) reflexive, antisymmetric, and not transitive.
d) reflexive, symmetric, and transitive.
e) reflexive, antisymmetric, and transitive.

4. a) How many reflexive relations are there on a set with n elements?
b) How many symmetric relations are there on a set with n elements?
c) How many antisymmetric relations are there on a set with n elements?

5. a) Explain how an n-ary relation can be used to represent information about students at a university.
b) How can the 5-ary relation containing names of students, their addresses, telephone numbers, majors, and grade point averages be used to form a 3-ary relation containing the names of students, their majors, and their grade point averages?
c) How can the 4-ary relation containing names of students, their addresses, telephone numbers, and majors and the 4-ary relation containing names of students, their student numbers, majors, and numbers of credit hours, be combined into a single n-ary relation?

6. a) Explain how to use a zero–one matrix to represent a relation on a finite set.
b) Explain how to use the zero–one matrix representing a relation to determine whether the relation is reflexive, symmetric, and/or antisymmetric.

7. a) Explain how to use a directed graph to represent a relation on a finite set.
b) Explain how to use the directed graph representing a relation to determine whether a relation is reflexive, symmetric, and/or antisymmetric.

8. a) Define the reflexive closure and the symmetric closure of a relation.
b) How can you construct the reflexive closure of a relation?
c) How can you construct the symmetric closure of a relation?
d) Find the reflexive closure and the symmetric closure of the relation $\{(1, 2), (2, 3), (2, 4), (3, 1)\}$ on the set $\{1, 2, 3, 4\}$.

9. a) Define the transitive closure of a relation.
b) Can the transitive closure of a relation be obtained by including all pairs (a, c) such that (a, b) and (b, c) belong to the relation?
c) Describe two algorithms for finding the transitive closure of a relation.
d) Find the transitive closure of the relation $\{(1,1), (1,3), (2,1), (2,3), (2,4), (3,2), (3,4), (4,1)\}$.

10. a) Define an equivalence relation.
b) Which relations on the set $\{a, b, c, d\}$ are equivalence relations and contain (a, b) and (b, d)?

11. a) Show that congruence modulo m is an equivalence relation whenever m is a positive integer.
b) Show that the relation $\{(a, b) \mid a \equiv \pm b \pmod 7\}$ is an equivalence relation on the set of integers.

12. a) What are the equivalence classes of an equivalence relation?
b) What are the equivalence classes of the congruent modulo 5 relation?
c) What are the equivalence classes of the equivalence relation in Question 11(b)?

13. Explain the relationship between equivalence relations on a set and partitions of this set.

14. a) Define a partial ordering.
b) Show that the divisibility relation on the set of positive integers is a partial order.

15. Explain how partial orderings on the sets A_1 and A_2 can be used to define a partial ordering on the set $A_1 \times A_2$.

16. a) Explain how to construct the Hasse diagram of a partial order on a finite set.
b) Draw the Hasse diagram of the divisibility relation on the set $\{2, 3, 5, 9, 12, 15, 18\}$.

17. a) Define a maximal element of a poset and the greatest element of a poset.
b) Give an example of a poset that has three maximal elements.
c) Give an example of a poset with a greatest element.

18. a) Define a lattice.
b) Give an example of a poset with five elements that is a lattice and an example of a poset with five elements that is not a lattice.

19. a) Show that every finite subset of a lattice has a greatest lower bound and a least upper bound.
b) Show that every lattice with a finite number of elements has a least element and a greatest element.

20. a) Define a well-ordered set.
b) Describe an algorithm for producing a well-ordered set from a partially ordered set.
c) Explain how the algorithm from (b) can be used to order the tasks in a project if each task can be done only after one or more of the other tasks have been completed.

Supplementary Exercises

1. Let S be the set of all strings of English letters. Determine whether the following relations are reflexive, irreflexive, symmetric, antisymmetric, and/or transitive.

a) $R_1 = \{(a, b) \mid a$ and b have no letters in common$\}$
b) $R_2 = \{(a, b) \mid a$ and b are not the same length$\}$
c) $R_3 = \{(a, b) \mid a$ is longer than $b\}$

2. Construct a relation on the set $\{a, b, c, d\}$ that is
a) reflexive, symmetric, but not transitive.
b) irreflexive, symmetric, and transitive.
c) irreflexive, antisymmetric, and not transitive.
d) reflexive, neither symmetric nor antisymmetric, transitive.
e) neither reflexive, irreflexive, symmetric, antisymmetric, nor transitive.

3. Show that the relation R on $\mathbf{Z} \times \mathbf{Z}$ defined by $(a, b) R (c, d)$ if and only if $a + d = b + c$ is an equivalence relation.

4. Show that a subset of an antisymmetric relation is also antisymmetric.

5. Let R be a reflexive relation on a set A. Show that $R \subseteq R^2$.

6. Suppose that R_1 and R_2 are reflexive relations on a set A. Show that $R_1 \oplus R_2$ is irreflexive.

7. Suppose that R_1 and R_2 are reflexive relations on a set A. Is $R_1 \cap R_2$ also reflexive? Is $R_1 \cup R_2$ also reflexive?

8. Suppose that R is a symmetric relation on a set A. Is $\overline{R}$ also symmetric?

9. Let R_1 and R_2 be symmetric relations. Is $R_1 \cap R_2$ also symmetric? Is $R_1 \cup R_2$ also symmetric?

10. A relation R is called **circular** if $a R b$ and $b R c$ imply that $c R a$. Show that R is reflexive and circular if and only if it is an equivalence relation.

11. Show that a primary key in an n-ary relation is a primary key in any projection of this relation that contains this key as one of its fields.

12. Is the primary key in an n-ary relation also a primary key in a larger relation obtained by taking the join of this relation with a second relation?

13. Show that the reflexive closure of the symmetric closure of a relation is the same as the symmetric closure of its reflexive closure.

14. Let R be the relation on the set of all mathematicians that contains the ordered pair (a, b) if and only if a and b have written a paper together.
a) Describe the relation R^2.
b) Describe the relation R^*.
web c) The **Erdős number** of a mathematician is 1 if this mathematician wrote a paper with the prolific Hungarian mathematician Paul Erdős, it is 2 if this mathematician did not write a joint paper with Erdős but wrote a joint paper with someone who wrote a joint paper with Erdős, and so on (except that the Erdős number of Erdős himself is 0). Give a definition of the Erdős number in terms of paths in R.

***15. a)** Give an example to show that the transitive closure of the symmetric closure of a relation is not necessarily the same as the symmetric closure of the transitive closure of this relation.

---

web **Paul Erdos (1913–1996).** Paul Erdős, born in Budapest, Hungary, was the son of two high school mathematics teachers. He was a child prodigy; at age 3 he could multiply three-digit numbers in his head, and at 4 he discovered negative numbers on his own. Because his mother did not want to expose him to contagious diseases, he was mostly home-schooled. At 17 Erdős entered Eötvös University, graduating 4 years later with a Ph.D. in mathematics. After graduating he spent 4 years at Manchester, England, on a postdoctoral fellowship. In 1938 he went to the United States because of the difficult political situation in Hungary, especially for Jews. He spent much of his time in the United States, except for 1954 to 1962, when he was banned as part of the paranoia of the McCarthy era. He also spent considerable time in Israel.

Erdős made many significant contributions to combinatorics and to number theory. One of the discoveries of which he was most proud is his elementary proof (in the sense that it does not use any complex analysis) of the prime number theorem, which provides an estimate for the number of primes not exceeding a fixed positive integer. He also participated in the modern development of Ramsey theory.

Erdos traveled extensively throughout the world to work with other mathematicians, visiting conferences, universities, and research laboratories. He almost entirely devoted himself to mathematics, traveling from one mathematician to the next, proclaiming "My brain is open." Erdős was the author or coauthor of almost 1500 papers and had almost 500 coauthors. Since he had no permanent home, copies of these articles are kept by Ron Graham, a famous discrete mathematician at AT&T Laboratories, with whom he collaborated extensively and who took care of many of his worldly needs.

Erdős offered rewards, ranging from \$10 to \$10,000, for the solution of problems that he found particularly interesting, with the size of the reward depending on the difficulty of the problem. He paid out close to \$4000. Erdős had his own special language, using such terms as "epsilon" (child), "boss" (woman), "slave" (man), "captured" (married), "liberated" (divorced), "Supreme Fascist" (God), "Sam" (United States), and "Joe" (Soviet Union). Although he was curious about many things, he concentrated almost all his energy on mathematical research. He had no hobbies and no full-time JOB. He never married and apparently remained celibate. Erdős was extremely generous, donating much of the money he collected from prizes, awards, and stipends for scholarships and to worthwhile causes. He traveled extremely lightly and did not like having many material possessions.

b) Show, however, that the transitive closure of the symmetric closure of a relation must contain the symmetric closure of the transitive closure of this relation.

16. **a)** Let S be the set of subroutines of a computer program. Define the relation R by **P** R **Q** if subroutine **P** calls subroutine **Q** during its execution. Describe the transitive closure of R.

b) For which subroutines **P** does (**P**, **P**) belong to the transitive closure of R?

c) Describe the reflexive closure of the transitive closure of R.

17. Suppose that R and S are relations on a set A with $R \subseteq S$ such that the closures of R and S with respect to a property **P** both exist. Show that the closure of R with respect to **P** is a subset of the closure of S with respect to **P**.

18. Show that the symmetric closure of the union of two relations is the union of their symmetric closures.

***19.** Devise an algorithm, based on the concept of interior vertices, that finds the length of the longest path between two vertices in a directed graph, or determines that there are arbitrarily long paths between these vertices.

20. Which of the following are equivalence relations on the set of all people?

a) $\{(x, y) \mid x$ and y have the same sign of the zodiac$\}$

b) $\{(x, y) \mid x$ and y were born in the same year$\}$

c) $\{(x, y) \mid x$ and y have been in the same city$\}$

***21.** How many different equivalence relations with exactly three different equivalence classes are there on a set with five elements?

22. Show that $\{(x, y) \mid x - y \in \mathbf{Q}\}$ is an equivalence relation on the set of real numbers where **Q** denotes the set of rational numbers. What are $[1]$, $[\frac{1}{2}]$, and $[\pi]$?

23. Suppose that $P_1 = \{A_1, A_2, \ldots, A_m\}$ and $P_2 = \{B_1, B_2, \ldots, B_n\}$ are both partitions of the set S. Show that the collection of nonempty subsets of the form $A_i \cap B_j$ is a partition of S that is a refinement of both P_1 and P_2 (see the preamble to Exercise 27 of Section 6.5).

***24.** Show that the transitive closure of the symmetric closure of the reflexive closure of a relation R is the smallest equivalence relation that contains R.

25. Let $\mathbf{R}(S)$ be the set of all relations on a set S. Define the relation $\preccurlyeq$ on $\mathbf{R}(S)$ by $R_1 \preccurlyeq R_2$ if $R_1 \subseteq R_2$, where R_1 and R_2 are relations on S. Show that $(\mathbf{R}(S), \preccurlyeq)$ is a poset.

26. Let $\mathbf{P}(S)$ be the set of all partitions of the set S. Define the relation $\preccurlyeq$ on $\mathbf{P}(S)$ by $P_1 \preccurlyeq P_2$ if P_1 is a refinement of P_2 (see Exercise 27 of Section 6.5). Show that $(\mathbf{P}(S), \preccurlyeq)$ is a poset.

27. Find an ordering of the tasks of a software project if the Hasse diagram for the tasks of this project is as shown on the facing page.

A subset of a poset such that every two elements of this subset are comparable is called a **chain.** A subset of a poset is called an **antichain** if every two elements of this subset are incomparable.

28. Find all chains in the posets with the Hasse diagrams shown in Exercises 17–19 in Section 6.6.

29. Find all antichains in the posets with the Hasse diagrams shown in Exercises 17–19 in Section 6.6.

30. Find an antichain with the greatest number of elements in the poset with the Hasse diagram of Exercise 24 in Section 6.6.

31. Show that every maximal chain in a finite poset $(S, \preccurlyeq)$ contains a minimal element of S. (A maximal chain is a chain that is not a subset of a larger chain.)

****32.** Show that a poset can be partitioned into k chains, where k is the largest number of elements in an antichain in this poset.

***33.** Show that in any group of $mn+1$ people there is either a list of $m+1$ people where a person in the list (except for the first person listed) is a descendant of the previous person on the list, or there are $n+1$ people such that none of these people is a descendant of any of the other n people. (*Hint:* Use Exercise 32.)

***34.** Establish the **generalized induction principle:** $P(x)$ is true for every element x in a well-ordered set S if $P(x_0)$ is true, where x_0 is the least element of S (the basis case), and if $P(x)$ is true for all $x \prec y$, then $P(y)$ is true (the inductive step).

35. Use the generalized induction principle on the well-ordered set $(\mathbf{Z}^+ \cup \{0\} \times \mathbf{Z}^+ \cup \{0\})$ (with lexicographic ordering) to show that $a_{m,n} = [n(n+1)/2] + m$, where $a_{0,0} = 0$ and

$$a_{m,n} = \begin{cases} a_{m-1,n} + 1 & \text{if } n = 0, \\ a_{m,n-1} + n & \text{if } n \neq 0. \end{cases}$$

A relation R on a set A is a **quasi-ordering** on A if R is reflexive and transitive.

36. Let R be the relation on the set of all functions from $\mathbf{Z}^+$ to $\mathbf{Z}^+$ such that (f, g) belongs to R if and only if f is $O(g)$. Show that R is a quasi-ordering.

37. Let R be a quasi-ordering on a set A. Show that $R \cap R^{-1}$ is an equivalence relation.

***38.** Let R be a quasi-ordering and let S be the relation on the set of equivalence classes of $R \cap R^{-1}$ such that (C, D) belongs to S, where C and D are equivalence classes of R, if and only if there are elements c of C and d of D such that (c, d) belongs to R. Show that S is a partial ordering.

Let L be a lattice. Define the **meet** $(\wedge)$ and **join** $(\vee)$ operations by $x \wedge y = \text{glb}(x, y)$ and $x \vee y = \text{lub}(x, y)$.

39. Show that the following properties hold for all elements x, y, and z of a lattice L.

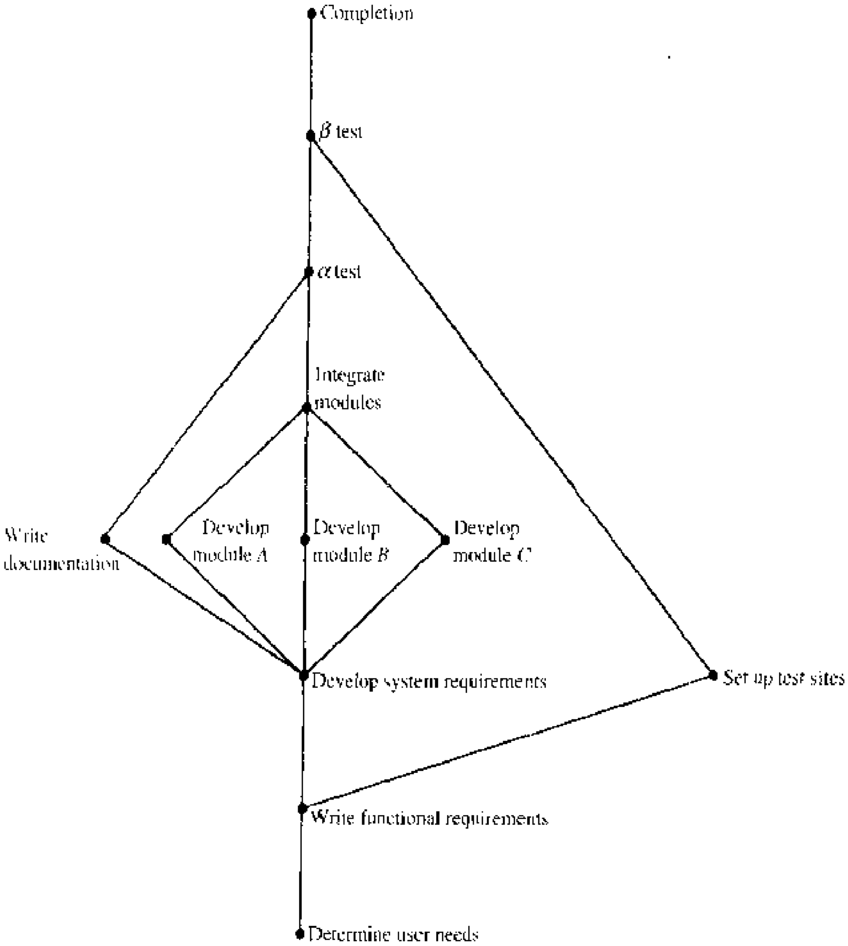

Completion
β test
α test
Integrate modules
Write documentation
Develop module A
Develop module B
Develop module C
Develop system requirements
Set up test sites
Write functional requirements
Determine user needs

Figure for Exercise 27.

a) $x \wedge y = y \wedge x$ and $x \vee y = y \vee x$ (**commutative laws**)

b) $(x \wedge y) \wedge z = x \wedge (y \wedge z)$ and $(x \vee y) \vee z = x \vee (y \vee z)$ (**associative laws**)

c) $x \wedge (x \vee y) = x$ and $x \vee (x \wedge y) = x$ (**absorption laws**)

d) $x \wedge x = x$ and $x \vee x = x$ (**idempotent laws**)

40. Show that if x and y are elements of a lattice L, then $x \vee y = y$ if and only if $x \wedge y = x$.

A lattice L is **bounded** if it has both an **upper bound**, denoted by 1, such that $x \preccurlyeq 1$ for all $x \in L$ and a **lower bound**, denoted by 0, such that $0 \preccurlyeq x$ for all $x \in L$.

41. Show that if L is a bounded lattice with upper bound 1 and lower bound 0 then the following properties hold *for all elements* $x \in L$.

a) $x \vee 1 = 1$ **b)** $x \wedge 1 = x$

c) $x \vee 0 = x$ **d)** $x \wedge 0 = 0$

42. Show that every finite lattice is bounded.

A lattice is called **distributive** if $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$ and $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ for all x, y, and z in L.

***43.** Give an example of a lattice which is not distributive.

44. Show that the lattice $(P(S), \subseteq)$ where $P(S)$ is the power set of a finite set S is distributive.

45. Is the lattice $(\mathbf{Z}^+, |)$ distributive?

The **complement** of an element a of a bounded lattice L with upper bound 1 and lower bound 0 is an element b such that $a \vee b = 1$ and $a \wedge b = 0$. Such a lattice is **complemented** if every element of the lattice has a complement.

46. Give an example of a finite lattice where at least one element has more than one complement and at least one element has no complement.

47. Show that the lattice $(P(S), \subseteq)$ where $P(S)$ is the power set of a finite set S is complemented.

***48.** Show that if L is a finite distributive lattice, then an element of L has at most one complement.

Computer Projects

WRITE PROGRAMS WITH THE FOLLOWING INPUT AND OUTPUT.

1. Given the matrix representing a relation on a finite set, determine whether the relation is reflexive and/or irreflexive.
2. Given the matrix representing a relation on a finite set, determine whether the relation is symmetric and/or antisymmetric.
3. Given the matrix representing a relation on a finite set, determine whether the relation is transitive.
4. Given a positive integer n, display all the relations on a set with n elements.

*5. Given a positive integer n, determine the number of transitive relations on a set with n elements.

*6. Given a positive integer n, determine the number of equivalence relations on a set with n elements.

*7. Given a positive integer n, display all the equivalence relations on the set of the n smallest positive integers.

8. Given an n-ary relation, find the projection of this relation when specified fields are deleted.
9. Give an m-ary relation and an n-ary relation, and a set of common fields, find the join of these relations with respect to these common fields.
10. Given the matrix representing a relation on a finite set, find the matrix representing the reflexive closure of this relation.
11. Given the matrix representing a relation on a finite set, find the matrix representing the symmetric closure of this relation.
12. Given the matrix representing a relation on a finite set, find the matrix representing the transitive closure of this relation by computing the join of the powers of the matrix representing the relation.
13. Given the matrix representing a relation on a finite set, find the matrix representing the transitive closure of this relation using Warshall's algorithm.
14. Given the matrix representing a relation on a finite set, find the matrix representing the smallest equivalence relation containing this relation.
15. Given a partial ordering on a finite set, find a total ordering compatible with it using topological sorting.

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. Display all the different relations on a set with four elements.
2. Display all the different reflexive and symmetric relations on a set with six elements.
3. Display all the reflexive and transitive relations on a set with five elements.

*4. Determine how many transitive relations there are on a set with n elements for all positive integers n with $n \leq 7$.

5. Find the transitive closure of a relation of your choice on a set with at least 20 elements. Either use a relation that corresponds to direct links in a particular transportation or communications network or use a randomly generated relation.
6. Compute the number of different equivalence relations on a set with n elements for all positive integers n not exceeding 20.
7. Display all the equivalence relations on a set with seven elements.

*8. Display all the partial orders on a set with five elements.

*9. Display all the lattices on a set with five elements.

Writing Projects

RESPOND TO THE FOLLOWING QUESTIONS WITH ESSAYS USING OUTSIDE SOURCES.

1. Discuss the concept of a fuzzy relation. How are fuzzy relations used?
2. Describe the basic principles of relational databases, going beyond what was covered in Section 6.2. How widely used are relational databases as compared with other types of databases?
3. Look up the original papers by Warshall and by Roy (in French) in which they develop algorithms for finding

transitive closures. Discuss their approaches. Why do you suppose that what we call Warshall's algorithm was discovered independently by more than one person?

4. Describe how equivalence classes can be used to define the rational numbers as classes of pairs of integers and how the basic arithmetic operations on rational numbers can be defined following this approach. (See Exercise 10 in Section 6.5.)
5. Explain how Helmut Hasse used what we now call Hasse diagrams.
6. Describe some of the mechanisms used to enforce information flow policies in computer operating systems.
7. Discuss the use of the Program Evaluation and Review Technique (PERT) to schedule the tasks of a large complicated project. How widely is PERT used?
8. Discuss the use of the Critical Path Method (CPM) to find the shortest time for the completion of a project. How widely is CPM used?
9. Discuss the concept of *duality* in a lattice. Explain how duality can be used to establish new results.
10. Explain what is meant by a *modular lattice*. Describe some of the properties of modular lattices and describe how modular lattices arise in the study of projective geometry.

7 Graphs

Graph theory is an old subject with many modern applications. Its basic ideas were introduced in the eighteenth century by the great Swiss mathematician Leonhard Euler. He used graphs to solve the famous Königsberg bridge problem, which we will discuss in this chapter.

Graphs are used to solve problems in many fields. For instance, graphs can be used to determine whether a circuit can be implemented on a planar circuit board. We can distinguish between two chemical compounds with the same molecular formula but different structures using graphs. We can determine whether two computers are connected by a communications link using graph models of computer networks. Graphs with weights assigned to their edges can be used to solve problems such as finding the shortest path between two cities in a transportation network. We can also use graphs to schedule exams and assign channels to television stations.

7.1

Introduction to Graphs

web

Graphs are discrete structures consisting of vertices and edges that connect these vertices. There are several different types of graphs that differ with respect to the kind and number of edges that can connect a pair of vertices. Problems in almost every conceivable discipline can be solved using graph models. We will give examples to show how graphs are used as models in a variety of areas. For instance, we will show how graphs are used to represent the competition of different species in an ecological niche, how graphs are used to represent who influences whom in an organization, and how graphs are used to represent the outcome of tournaments. Later we will show how graphs can be used to solve many types of problems, such as computing the number of different combinations of flights between two cities in an airline network, determining whether it is possible to walk down all the streets in a city without going down a street twice, and finding the number of colors needed to color the regions of a map.

TYPES OF GRAPHS

We will introduce the different types of graphs by showing how each can be used to model a computer network. Suppose that a network is made up of computers and telephone lines between computers. We can represent the location of each computer by a point and each telephone line by an arc, as shown in Figure 1.

We make the following observations about the network in Figure 1. There is at most one telephone line between two computers in this network, each line operates in both directions, and no computer has a telephone line to itself. Consequently this network can be modeled using a **simple graph**, consisting of vertices that represent

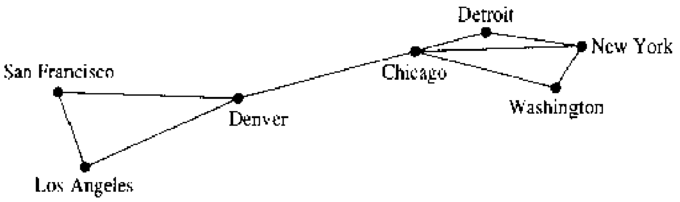


FIGURE 1 A Computer Network.

the computers and undirected edges that represent telephone lines, where each edge connects two distinct vertices and no two edges connect the same pair of vertices.

DEFINITION 1. **A *simple graph* $G = (V, E)$ consists of V, a nonempty set of *vertices*, and E, a set of unordered pairs of distinct elements of V called *edges*.**

Sometimes there are multiple telephone lines between computers in a network. This is the case when there is heavy traffic between computers. A network with multiple lines is displayed in Figure 2. Simple graphs are not sufficient to model such networks. Instead, **multigraphs** are used, which consist of vertices and undirected edges between these vertices, with multiple edges between pairs of vertices allowed. Every simple graph is also a multigraph. However, not all multigraphs are simple graphs, since in a multigraph two or more edges may connect the same pair of vertices.

We cannot use a pair of vertices to specify an edge of a graph when multiple edges are present. This makes the formal definition of multigraphs somewhat complicated.

DEFINITION 2. **A *multigraph* $G = (V, E)$ consists of a set V of vertices, a set E of edges, and a function f from E to $\{\{u, v\} \mid u, v \in V, u \neq v\}$. The edges e_1 and e_2 are called *multiple* or *parallel edges* if $f(e_1) = f(e_2)$.**

A computer network may contain a telephone line from a computer to itself (perhaps for diagnostic purposes). Such a network is shown in Figure 3. We cannot use multigraphs to model such networks, since **loops,** which are edges from a vertex to itself, are not allowed in multigraphs. Instead, **pseudographs** are used. Pseudographs are more general than multigraphs, since an edge in a pseudograph may connect a vertex with itself.

To formally define pseudograph we must be able to associate edges to sets containing just one vertex.

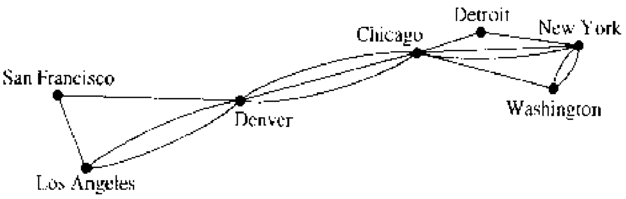


FIGURE 2 A Computer Network with Multiple Lines.

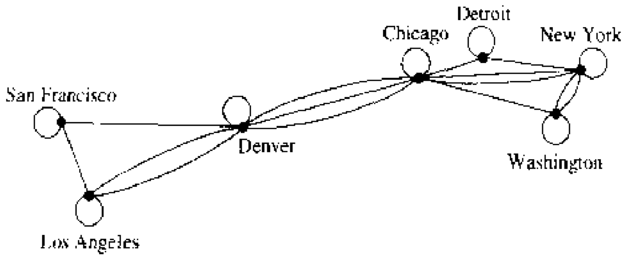


FIGURE 3 **A Computer Network with Diagnostic Lines.**

DEFINITION 3. **A *pseudograph* $G = (V, E)$ consists of a set V of vertices, a set E of edges, and a function f from E to $\{\{u, v\} \mid u, v \in V\}$. An edge is a *loop* if $f(e) = \{u, u\} = \{u\}$ for some $u \in V$.**

The reader should note that multiple edges in a pseudograph are associated to the same pair of vertices. However, we will say that $\{u, v\}$ is an edge of a graph $G = (V, E)$ if there is at least one edge e with $f(e) = \{u, v\}$. We will not distinguish between the edge e and the set $\{u, v\}$ associated to it unless the identity of individual multiple edges is important.

To summarize, pseudographs are the most general type of undirected graphs since they may contain loops and multiple edges. Multigraphs are undirected graphs that may contain multiple edges but may not have loops. Finally, simple graphs are undirected graphs with no multiple edges or loops.

The telephone lines in a computer network may not operate in both directions. For instance, in Figure 4 the host computer in New York can only receive data from other computers and cannot send out data. The other telephone lines operate in both directions and are represented by pairs of edges in opposite directions.

We use directed graphs (which were studied in Chapter 6) to model such networks. The edges of a directed graph are ordered pairs. Loops, ordered pairs of the same element, are allowed, but multiple edges in the same direction between two vertices are not. Recall the following definition.

DEFINITION 4. **A *directed graph* (V, E) consists of a set of vertices V and a set of edges E that are ordered pairs of elements of V.**

Finally, multiple lines may be present in the computer network, so that there may be several one-way lines to the host in New York from each location, and perhaps more

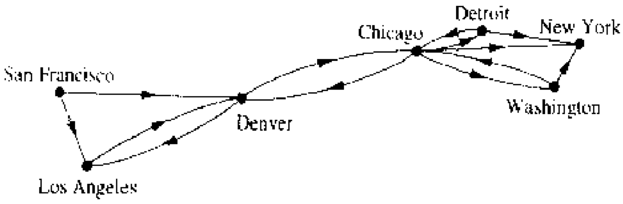


FIGURE 4 **A Communications Network with One-Way Telephone Lines.**

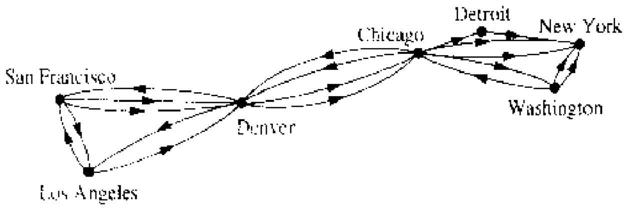


FIGURE 5 **A Computer Network with Multiple One-Way Lines.**

than one line back to each remote computer from the host. Such a network is shown in Figure 5. Directed graphs are not sufficient for modeling such a network, since multiple edges are not allowed in these graphs. Instead, **directed multigraphs,** which may have multiple directed edges from a vertex to a second (possibly the same) vertex, are needed. The formal definition of a directed multigraph follows.

DEFINITION 5. **A *directed multigraph* $G = (V, E)$ consists of a set V of vertices, a set E of edges, and a function f from E to $\{(u, v) \mid u, v \in V\}$. The edges e_1 and e_2 are *multiple edges* if $f(e_1) = f(e_2)$.**

The reader should note that multiple directed edges are associated to the same pair of vertices. However, we will say that (u, v) is an edge of $G = (V, E)$ as long as there is at least one edge e with $f(e) = (u, v)$. We will not make the distinction between the edge e and the ordered pair (u, v) associated to it unless the identity of individual multiple edges is important.

This terminology for the various types of graphs makes clear whether the edges of a graph are associated to ordered or unordered pairs, whether multiple edges are allowed, and whether loops are allowed. We will use **graph** to describe graphs with directed or undirected edges, with or without loops and multiple edges. We will use the terms **undirected graph** or **pseudograph** for an undirected graph that may have multiple edges and loops. We will always use the adjective **directed** when referring to graphs that have ordered pairs associated to their edges. The definitions of the various types of graphs are summarized in Table 1. Because of the relatively modern interest in graph theory, and because it has applications to a wide variety of disciplines, many different terminologies of graph theory are commonly used. The reader should determine how such terms are being used whenever they are encountered. Perhaps this terminology will become standardized someday.

TABLE 1 **Graph Terminology.**

Type	*Edges*	*Multiple Edges Allowed?*	*Loops Allowed?*
Simple graph	Undirected	No	No
Multigraph	Undirected	Yes	No
Pseudograph	Undirected	Yes	Yes
Directed graph	Directed	No	Yes
Directed multigraph	Directed	Yes	Yes

GRAPH MODELS

web

Graphs are used in a wide variety of models. We will present a few graph models from diverse fields here. Others will be introduced in subsequent sections of this and the following chapters.

EXAMPLE 1 Niche Overlap Graphs in Ecology Graphs are used in many models involving the interaction of different species of animals. For instance, the competition between species in an ecosystem can be modeled using a **niche overlap graph.** Each species is represented by a vertex. An undirected edge connects two vertices if the two species represented by these vertices compete (that is, some of the food resources they use are the same). The graph in Figure 6 models the ecosystem of a forest. We see from this graph that squirrels and raccoons compete but that crows and shrews do not. ■

EXAMPLE 2 Influence Graphs In studies of group behavior it is observed that certain people can influence the thinking of others. A directed graph called an **influence graph** can be used to model this behavior. Each person of the group is represented by a vertex. There is a directed edge from vertex a to vertex b when the person represented by vertex a influences the person represented by vertex b. An example of an influence graph for members of a group is shown in Figure 7. In the group modeled by this influence graph, Deborah can influence Brian, Fred, and Linda, but no one can influence her. Also, Yvonne and Brian can influence each other. ■

EXAMPLE 3 Round-Robin Tournaments A tournament where each team plays each other team exactly once is called a **round-robin tournament**. Such tournaments can be modeled using directed graphs where each team is represented by a vertex. Note that (a, b) is an edge if team a beats team b. Such a directed graph model is presented in Figure 8. Note that Team 1 is undefeated in this tournament, and Team 3 is winless. ■

EXAMPLE 4 Precedence Graphs and Concurrent Processing Computer programs can be executed more rapidly by executing certain statements concurrently. It is important not to

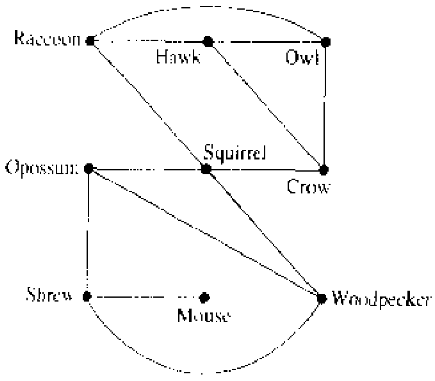


FIGURE 6 A Niche Overlap Graph.

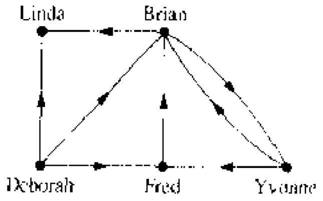


FIGURE 7 An Influence Graph.

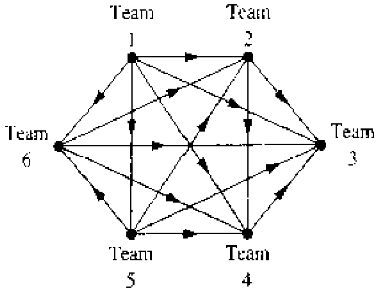


FIGURE 8 A Graph Model of a Round-Robin Tournament.

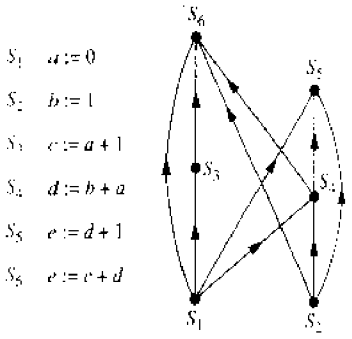


FIGURE 9 A Precedence Graph.

execute a statement that requires results of statements not yet executed. The dependence of statements on previous statements can be represented by a directed graph. Each statement is represented by a vertex, and there is an edge from one vertex to a second vertex if the statement represented by the second vertex cannot be executed before the statement represented by the first vertex has been executed. This graph is called a **precedence graph.** A computer program and its graph are displayed in Figure 9. For instance, the graph shows that statement S_5 cannot be executed before statements S_1, S_2, and S_4 are executed. ■

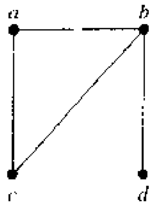
Exercises

1. Draw graph models, stating the type of graph used, to represent airline routes where every day there are four flights from Boston to Newark, two flights from Newark to Boston, three flights from Newark to Miami, two flights from Miami to Newark, one flight from Newark to Detroit, two flights from Detroit to Newark, three flights from Newark to Washington, two flights from Washington to Newark, and one flight from Washington to Miami, with
   a) an edge between vertices representing cities that have a flight between them (in either direction).
   b) an edge between vertices representing cities for each flight that operates between them (in either direction).
   c) an edge between vertices representing cities for each flight that operates between them (in either direction), plus a loop for a special sightseeing trip that takes off and lands in Miami.
   d) an edge from a vertex representing a city where a flight starts to the vertex representing the city where it ends.
   e) an edge for each flight from a vertex representing a city where the flight begins to the vertex representing the city where the flight ends.

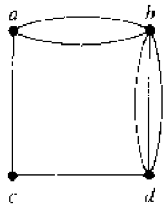
2. What kind of graph can be used to model a highway system between major cities where
   a) there is an edge between the vertices representing cities if there is an interstate highway between them?
   b) there is an edge between the vertices representing cities for each interstate highway between them?
   c) there is an edge between the vertices representing cities for each interstate highway between them, and there is a loop at the vertex representing a city if there is an interstate highway that circles this city?

For Exercises 3–9, determine whether the graph shown is a simple graph, a multigraph (and not a simple graph), a pseudograph (and not a multigraph), a directed graph, or a directed multigraph (and not a directed graph).

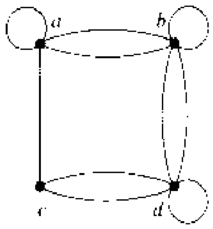
3.



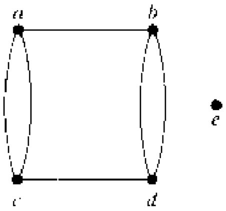
4.



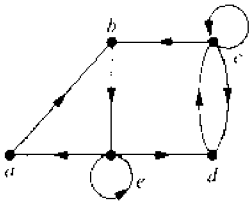
5.



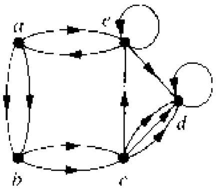
6.



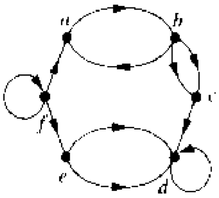
7.



8.



9.



10. For each undirected graph in Exercises 3–9 that is not simple, find a set of edges to remove to make it simple.

11. The **intersection graph** of a collection of sets $A_1, A_2, \ldots, A_n$ is the graph that has a vertex for each of these sets and has an edge connecting the vertices representing two sets if these sets have a nonempty intersection. Construct the intersection graph of the following collections of sets.

a) $A_1 = \{0, 2, 4, 6, 8\}$, $A_2 = \{0, 1, 2, 3, 4\}$,
$A_3 = \{1, 3, 5, 7, 9\}$, $A_4 = \{5, 6, 7, 8, 9\}$,
$A_5 = \{0, 1, 8, 9\}$

b) $A_1 = \{\ldots, -4, -3, -2, -1, 0\}$,
$A_2 = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$,
$A_3 = \{\ldots, -6, -4, -2, 0, 2, 4, 6, \ldots\}$,
$A_4 = \{\ldots, -5, -3, -1, 1, 3, 5, \ldots\}$,
$A_5 = \{\ldots, -6, -3, 0, 3, 6, \ldots\}$

c) $A_1 = \{x \mid x < 0\}$,
$A_2 = \{x \mid -1 < x < 0\}$,
$A_3 = \{x \mid 0 < x < 1\}$,
$A_4 = \{x \mid -1 < x < 1\}$,
$A_5 = \{x \mid x > -1\}$,
$A_6 = \mathbf{R}$.

12. Use the niche overlap graph in Figure 6 to determine the species that compete with hawks.

13. Construct a niche overlap graph for six species of birds where the hermit thrush competes with the robin and with the blue jay, the robin also competes with the mockingbird, the mockingbird also competes with the blue jay, and the nuthatch competes with the hairy woodpecker.

14. Who can influence Fred and whom can Fred influence in the influence graph in Example 2?

15. Construct an influence graph for the board members of a company if the President can influence the Director of Research and Development, the Director of Marketing, and the Director of Operations; the Director of Research and Development can influence the Director of Operations; the Director of Marketing can influence the Director of Operations; and no one can influence, or be influenced by, the Chief Financial Officer.

16. Which other teams did Team 4 beat and which teams beat Team 4 in the round-robin tournament represented by the graph in Figure 8?

17. In a round-robin tournament the Tigers beat the Blue Jays, the Tigers beat the Cardinals, the Tigers beat the Orioles, the Blue Jays beat the Cardinals, the Blue Jays beat the Orioles, and the Cardinals beat the Orioles. Model this outcome with a directed graph.

18. Which statements must be executed before S_6 is executed in the program in Example 4? (Use the precedence graph in Figure 9.)

19. Construct a precedence graph for the following program:

$S_1 : x := 0$
$S_2 : x := x + 1$
$S_3 : y := 2$
$S_4 : z := y$
$S_5 : x := x + 2$
$S_6 : y := x + z$
$S_7 : z := 4$

20. Describe a discrete structure based on a graph that can be used to model airline routes and their flight times. (*Hint:* Add structure to a directed graph.)

21. Describe a discrete structure based on a graph that can be used to model relationships between pairs of individuals in a group, where each individual may either like, dislike, or be neutral about another individual, and the reverse relationship may be different. (*Hint:* Add structure to a directed graph. Treat separately the edges in opposite directions between vertices representing two individuals.)

7.2

Graph Terminology

INTRODUCTION

We introduce some of the basic vocabulary of graph theory in this section. We will use this vocabulary when we solve many different types of problems. One such problem involves determining whether a graph can be drawn in the plane so that no two of its edges cross. Another example is deciding whether there is a one-to-one correspondence between the vertices of two graphs that produces a one-to-one correspondence between the edges of the graphs. We will also introduce several important families of graphs often used as examples and in models.

BASIC TERMINOLOGY

First, we give some terminology that describes the vertices and edges of undirected graphs.

DEFINITION 1. **Two vertices u and v in an undirected graph G are called *adjacent* (or *neighbors*) in G if $\{u, v\}$ is an edge of G. If $e = \{u, v\}$, the edge e is called *incident with* the vertices u and v. The edge e is also said to *connect* u and v. The vertices u and v are called *endpoints* of the edge $\{u, v\}$.**

To keep track of how many edges are incident to a vertex, we make the following definition.

DEFINITION 2. **The *degree* of a vertex in an undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex. The degree of the vertex v is denoted by $\deg(v)$.**

EXAMPLE 1 What are the degrees of the vertices in the graphs G and H displayed in Figure 1?

Solution: In G, $\deg(a) = 2$, $\deg(b) = \deg(c) = \deg(f) = 4$, $\deg(d) = 1$, $\deg(e) = 3$, and $\deg(g) = 0$. In H, $\deg(a) = 4$, $\deg(b) = \deg(e) = 6$, $\deg(c) = 1$, and $\deg(d) = 5$. ■

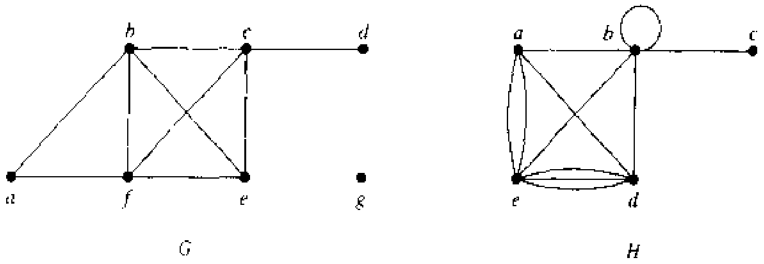


FIGURE 1 The Undirected Graphs G and H.

A vertex of degree 0 is called **isolated.** It follows that an isolated vertex is not adjacent to any vertex. Vertex g in graph G in Example 1 is isolated. A vertex is **pendant** if and only if it has degree 1. Consequently, a pendant vertex is adjacent to exactly one other vertex. Vertex d in graph G in Example 1 is pendant.

What do we get when we add the degrees of all the vertices of a graph $G = (V, E)$? Each edge contributes 2 to the sum of the degrees of the vertices since an edge is incident with exactly two (possibly equal) vertices. This means that the sum of the degrees of the vertices is twice the number of edges. We have the following result, which is sometimes called the Handshaking Theorem, because of the analogy between an edge having two endpoints and a handshake involving two hands.

THEOREM 1 The Handshaking Theorem **Let $G = (V, E)$ be an undirected graph with e edges. Then**

$$2e = \sum_{v \in V} \deg(v).$$

(Note that this applies even if multiple edges and loops are present.)

EXAMPLE 2 How many edges are there in a graph with 10 vertices each of degree 6?

Solution: Since the sum of the degrees of the vertices is $6 \cdot 10 = 60$, it follows that $2e = 60$. Therefore, $e = 30$. ■

Theorem 1 shows that the sum of the degrees of the vertices of an undirected graph is even. This simple fact has many consequences, one of which is given as Theorem 2.

THEOREM 2 **An undirected graph has an even number of vertices of odd degree.**

Proof: Let V_1 and V_2 be the set of vertices of even degree and the set of vertices of odd degree, respectively, in an undirected graph $G = (V, E)$. Then

$$2e = \sum_{v \in V} \deg(v) = \sum_{v \in V_1} \deg(v) + \sum_{v \in V_2} \deg(v).$$

Since $\deg(v)$ is even for $v \in V_1$, the first term in the right-hand side of the last equality is even. Furthermore, the sum of the two terms on the right-hand side of the last equality is even, since this sum is $2e$. Hence, the second term in the sum is also even. Since all the terms in this sum are odd, there must be an even number of such terms. Thus, there are an even number of vertices of odd degree. □

There is also some useful terminology for graphs with directed edges.

DEFINITION 3. **When (u, v) is an edge of the graph G with directed edges, u is** said to be ***adjacent to*** v and v is said to be ***adjacent from*** u. **The vertex u is called the** ***initial vertex*** of (u, v), and v is called the ***terminal*** or ***end vertex*** of (u, v). **The initial** vertex and terminal vertex of a loop are the same.

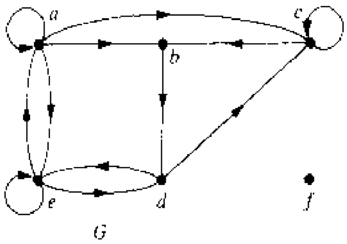


FIGURE 2 The Directed Graph G.

Since the edges in graphs with directed edges are ordered pairs, the definition of the degree of a vertex can be refined to reflect the number of edges with this vertex as the initial vertex and as the terminal vertex.

DEFINITION 4. In a graph with directed edges the *in-degree* of a vertex v, denoted by $\deg^-(v)$, is the number of edges with v as their terminal vertex. The *out-degree* of v, denoted by $\deg^+(v)$, is the number of edges with v as their initial vertex. (Note that a loop at a vertex contributes 1 to both the in-degree and the out-degree of this vertex.)

EXAMPLE 3 Find the in-degree and out-degree of each vertex in the graph G with directed edges shown in Figure 2.

Solution: The in-degrees in G are: $\deg^-(a) = 2$, $\deg^-(b) = 2$, $\deg^-(c) = 3$, $\deg^-(d) = 2$, $\deg^-(e) = 3$, and $\deg^-(f) = 0$. The out-degrees are: $\deg^+(a) = 4$, $\deg^+(b) = 1$, $\deg^+(c) = 2$, $\deg^+(d) = 2$, $\deg^+(e) = 3$, and $\deg^+(f) = 0$. ■

Since each edge has an initial vertex and a terminal vertex, the sum of the in-degrees and the sum of the out-degrees of all vertices in a graph with directed edges are the same. Both of these sums are the number of edges in the graph. This result is stated as the following theorem.

THEOREM 3 **Let $G = (V, E)$ be a graph with directed edges. Then**

$$\sum_{v \in V} \deg^-(v) = \sum_{v \in V} \deg^+(v) = |E|.$$

There are many properties of a graph with directed edges that do not depend on the direction of its edges. Consequently, it is often useful to ignore these directions. The undirected graph that results from ignoring directions of edges is called the **underlying undirected graph.** A graph with directed edges and its underlying undirected graph have the same number of edges.

SOME SPECIAL SIMPLE GRAPHS

We will now introduce several classes of simple graphs. These graphs are often used as examples and arise in many applications.

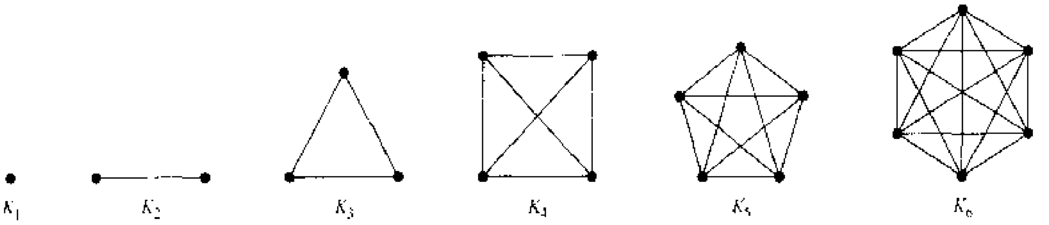


FIGURE 3 **The Graphs K_n, $1 \leq n \leq 6$.**

EXAMPLE 4 Complete Graphs The *complete graph* on n vertices, denoted by K_n, is the simple graph that contains exactly one edge between each pair of distinct vertices. The graphs K_n, for $n = 1, 2, 3, 4, 5, 6$, are displayed in Figure 3. ■

EXAMPLE 5 Cycles The *cycle* C_n, $n \geq 3$, consists of n vertices $v_1, v_2, \ldots, v_n$ and edges $\{v_1, v_2\}$, $\{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}$, and $\{v_n, v_1\}$. The cycles C_3, C_4, C_5, and C_6 are displayed in Figure 4. ■

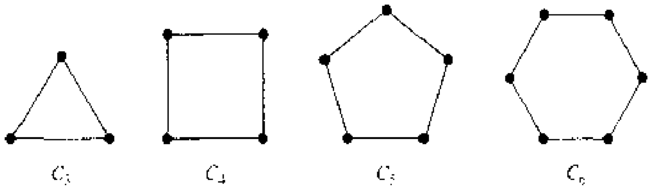


FIGURE 4 **The Cycles C_3, C_4, C_5, and C_6.**

EXAMPLE 6 Wheels We obtain the *wheel* W_n when we add an additional vertex to the cycle C_n, for $n \geq 3$, and connect this new vertex to each of the n vertices in C_n, by new edges. The wheels W_3, W_4, W_5, and W_6 are displayed in Figure 5. ■

EXAMPLE 7 n-Cubes The *n-cube*, denoted by Q_n, is the graph that has vertices representing the 2^n bit strings of length n. Two vertices are adjacent if and only if the bit strings that they represent differ in exactly one bit position. The graphs Q_1, Q_2, and Q_3 are displayed in Figure 6. ■

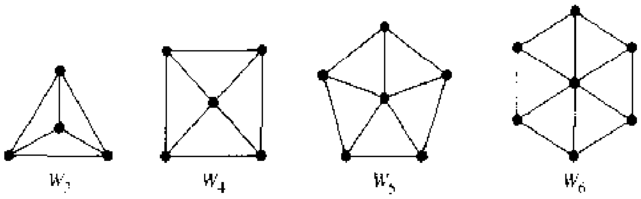


FIGURE 5 **The Wheels W_3, W_4, W_5, and W_6.**

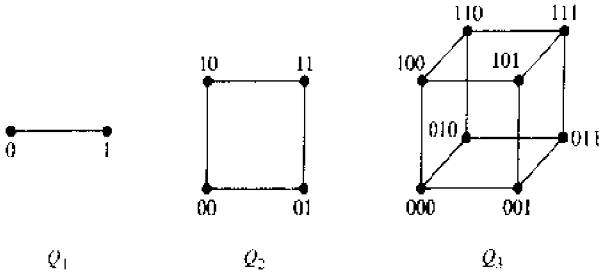


FIGURE 6 The n-cube Q_n for $n = 1, 2,$ and 3.

BIPARTITE GRAPHS

Sometimes a graph has the property that its vertex set can be divided into two disjoint subsets such that each edge connects a vertex in one of these subsets to a vertex in the other subset. For example, consider the graph representing marriages between people in a village, where each person is represented by a vertex and a marriage is represented by an edge. In this graph, each edge connects a vertex in the subset of vertices representing males and a vertex in the subset of vertices representing females. This leads us to the following definition.

DEFINITION 5. A simple graph G is called *bipartite* if its vertex set V can be partitioned into two disjoint nonempty sets V_1 and V_2 such that every edge in the graph connects a vertex in V_1 and a vertex in V_2 (so that no edge in G connects either two vertices in V_1 or two vertices in V_2).

In Example 8 we will show that C_6 is bipartite, and in Example 9 we will show that K_3 is not bipartite.

EXAMPLE 8 C_6 is bipartite, as shown in Figure 7, since its vertex set can be partitioned into the two sets $V_1 = \{v_1, v_3, v_5\}$ and $V_2 = \{v_2, v_4, v_6\}$, and every edge of C_6 connects a vertex in V_1 and a vertex in V_2. ■

EXAMPLE 9 K_3 is not bipartite. To see this, note that if we divide the vertex set of K_3 into two disjoint sets, one of the two sets must contain two vertices. If the graph were bipartite, these two vertices could not be connected by an edge, but in K_3 each vertex is connected to every other vertex by an edge. ■

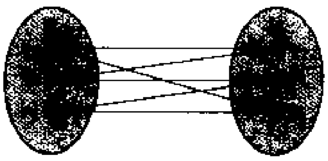


FIGURE 7 Showing That C_6 Is Bipartite.

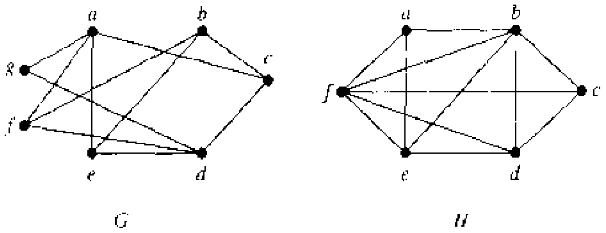


FIGURE 8 The Undirected Graphs G and H.

EXAMPLE 10 Are the graphs G and H displayed in Figure 8 bipartite?

Solution: Graph G is bipartite, since its vertex set is the union of two disjoint sets, $\{a, b, d\}$ and $\{c, e, f, g\}$, and each edge connects a vertex in one of these subsets to a vertex in the other subset. (Note that for G to be bipartite it is not necessary that every vertex in $\{a, b, d\}$ be adjacent to every vertex in $\{c, e, f, g\}$. For instance, b and g are not adjacent.)

Graph H is not bipartite since its vertex set cannot be partitioned into two subsets so that edges do not connect two vertices from the same subset. (The reader should verify this by considering the vertices a, b, and f.) ■

EXAMPLE 11 Complete Bipartite Graphs The *complete bipartite graph* $K_{m,n}$ is the graph that has its vertex set partitioned into two subsets of m and n vertices, respectively. There is an edge between two vertices if and only if one vertex is in the first subset and the other vertex is in the second subset. The complete bipartite graphs $K_{2,3}$, $K_{3,3}$, $K_{3,5}$, and $K_{2,6}$ are displayed in Figure 9. ■

SOME APPLICATIONS OF SPECIAL TYPES OF GRAPHS

We will show how special types of graphs are used in models for data communications and parallel processing.

EXAMPLE 12 Local Area Networks The various computers in a building, such as minicomputers and personal computers, as well as peripheral devices such as printers and plotters, can be

Web

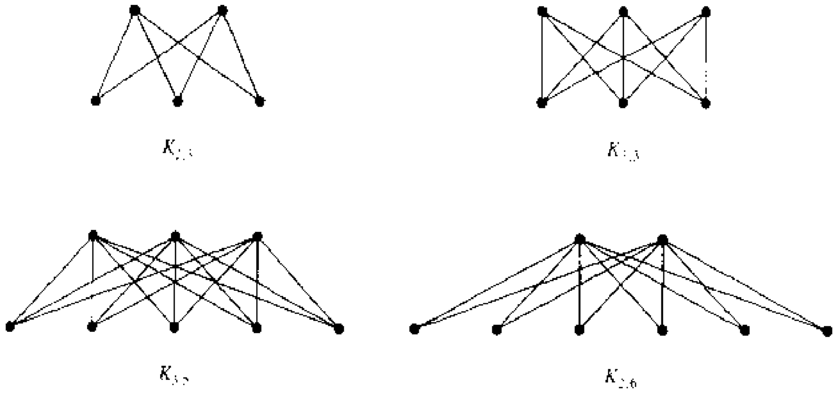


FIGURE 9 Some Complete Bipartite Graphs.

connected using a *local area network*. Some of these networks are based on a *star topology*, where all devices are connected to a central control device. A local area network can be represented using a complete bipartite graph $K_{1,n}$, as shown in Figure 10(a). Messages are sent from device to device through the central control device.

Other local area networks are based on a *ring topology*, where each device is connected to exactly two others. Local area networks with a ring topology are modeled using n-cycles, C_n, as shown in Figure 10(b). Messages are sent from device to device around the cycle until the intended recipient of a message is reached.

Finally, some local area networks use a hybrid of these two topologies. Messages may be sent around the ring, or through a central device. This redundancy makes the network more reliable. Local area networks with this redundancy can be modeled using wheels W_n, as shown in Figure 10(c). ◼

EXAMPLE 13 Interconnection Networks for Parallel Computation Until recently, computers executed programs one operation at a time. Consequently, the algorithms written to solve problems were designed to perform one step at a time; such algorithms are called **serial.** (Almost all algorithms described in this book are serial.) However, many computationally intense problems, such as weather simulations, medical imaging, and cryptanalysis, cannot be solved in a reasonable amount of time using serial operations, even on a supercomputer. Furthermore, there is a physical limit to how fast a computer can carry out basic operations, so that there will always be problems that cannot be solved in a reasonable length of time using serial operations.

Parallel processing, which uses computers made up of many separate processors, each with its own memory, helps overcome the limitations of serial computers. **Parallel algorithms,** which break a problem into a number of subproblems that can be solved concurrently, can then be devised to rapidly solve problems using a computer with multiple processors. In a parallel algorithm, a single instruction stream controls the execution of the algorithm, sending subproblems to different processors, and directs the input and output of these subproblems to the appropriate processors.

When parallel processing is used, one processor may need output generated by another processor. Consequently, these processors need to be interconnected. We can use the appropriate type of graph to represent the interconnection network of the processors in a computer with multiple processors. In the following discussion, we will describe the most commonly used types of interconnection networks for parallel processors. The type of interconnection network used to implement a particular parallel algorithm depends on the requirements for exchange of data between processors, the desired speed, and, of course, the available hardware.

The simplest, but most expensive, network-interconnecting processors include a two-way link between each pair of processors. This network can be represented by

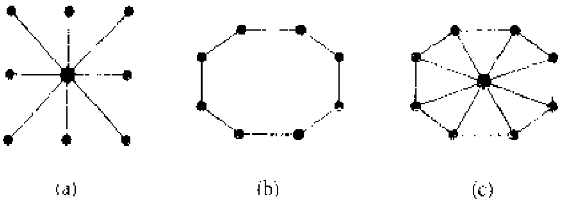


FIGURE 10 Star, Ring, and Hybrid Topologies for Local Area Networks.

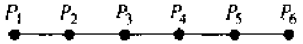


FIGURE 11 A Linear Array for Six Processors.

K_n, the complete graph on n vertices, when there are n processors. However, there are serious problems with this type of interconnection network because the required number of connections is so large. In reality, the number of direct connections to a processor is limited, so when there are a large number of processors, a processor cannot be linked directly to all others. For example, when there are 64 processors, $C(64, 2) = 2016$ connections would be required, and each processor would have to be directly connected to 63 others.

On the other hand, perhaps the simplest way to interconnect n processors is to use an arrangement known as a **linear array**. Each processor P_i, other than P_1 and P_n, is connected to its neighbors P_{i-1} and P_{i+1} via a two-way link. P_1 is connected only to P_2, and P_n is connected only to P_{n-1}. The linear array for six processors is shown in Figure 11. The advantage of a linear array is that each processor has at most two direct connections to other processors. The disadvantage is that it is sometimes necessary to use a large number of intermediate links, called **hops**, for processors to share information.

The **mesh network** (or **two-dimensional array**) is a commonly used interconnection network. In such a network, the number of processors is a perfect square, say $n = m^2$. The n processors are labeled $P(i, j)$, $0 \le i \le m - 1$, $0 \le j \le m - 1$. Two-way links connect processor $P(i, j)$ with its four neighbors, processors $P(i \pm 1, j)$ and $P(i, j \pm 1)$, as long as these are processors in the mesh. (Note that four processors, on the corners of the mesh, have only two adjacent processors, and other processors on the boundaries have only three neighbors. Sometimes a variant of a mesh network in which every processor has exactly four connections is used; see Exercise 44 at the end of this section.) The mesh network limits the number of links for each processor. Communication between some pairs of processors requires $O(\sqrt{n}) = O(m)$ intermediate links. (See Exercise 45 at the end of this section.) The graph representing the mesh network for 16 processors is shown in Figure 12.

One important type of interconnection network is the hypercube. For such a network, the number of processors is a power of 2, $n = 2^m$. The n processors are labeled $P_0, P_1, \ldots, P_{n-1}$. Each processor has two-way connections to m other processors. Processor P_i is linked to the processors with indices whose binary representations differ from the binary representation of i in exactly one bit. The hypercube network balances the number of direct connections for each processor and the number of intermediate connections required so that processors can communicate. Many computers have been built using a hypercube network, and many parallel algorithms have been

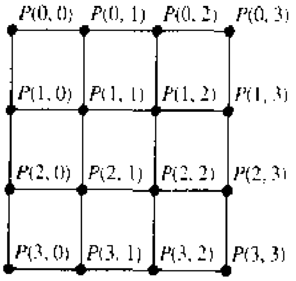


FIGURE 12 A Mesh Network for 16 Processors.

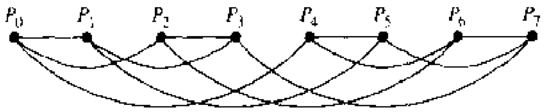


FIGURE 13 A Hypercube Network for Eight Processors.

devised that use a hypercube network. The graph Q_n, the n-cube, represents the hypercube network with n processors. Figure 13 displays the hypercube network for eight processors. (Figure 13 displays a different way to draw Q_3 than was shown in Figure 6.) ■

NEW GRAPHS FROM OLD

Sometimes we need only part of a graph to solve a problem. For instance, we may care only about the part of a large computer network that involves the computer centers in New York, Denver, Detroit, and Atlanta. Then we can ignore the other computer centers and all telephone lines not linking two of these specific four computer centers. In the graph model for the large network, we can remove the vertices corresponding to the computer centers other than the four of interest, and we can remove all edges incident with a vertex that was removed. When edges and vertices are removed from a graph, without removing endpoints of any remaining edges, a smaller graph is obtained. Such a graph is called a **subgraph** of the original graph.

DEFINITION 6. **A *subgraph* of a graph $G = (V, E)$ is a graph $H = (W, F)$ where $W \subseteq V$ and $F \subseteq E$.**

EXAMPLE 14 The graph G shown in Figure 14 is a subgraph of K_5. ■

Two or more graphs can be combined in various ways. The new graph that contains all the vertices and edges of these graphs is called the **union** of the graphs. We will give a more formal definition for the union of two simple graphs.

DEFINITION 7. **The *union* of two simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the simple graph with vertex set $V_1 \cup V_2$ and edge set $E_1 \cup E_2$. The union of G_1 and G_2 is denoted by $G_1 \cup G_2$.**

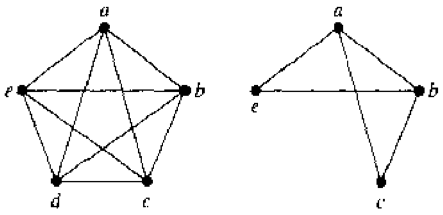
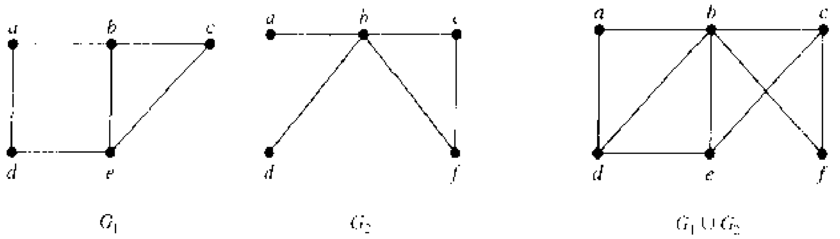


FIGURE 14 A Subgraph of K_5.



(a) (b)

FIGURE 15 (a) The Simple Graphs G_1 and G_2 and (b) Their Union $G_1 \cup G_2$.

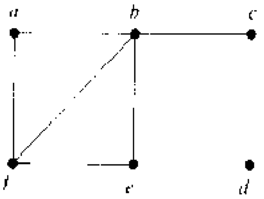
EXAMPLE 15 Find the union of the graphs G_1 and G_2 shown in Figure 15(a).

Solution: The vertex set of the union $G_1 \cup G_2$ is the union of the two vertex sets, namely, $\{a, b, c, d, e, f\}$. The edge set of the union is the union of the two edge sets. The union is displayed in Figure 15(b). ■

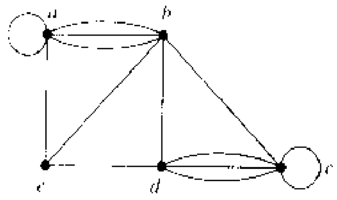
Exercises

In Exercises 1–3 find the number of vertices, the number of edges, and the degree of each vertex in the given undirected graph. Identify all isolated and pendant vertices.

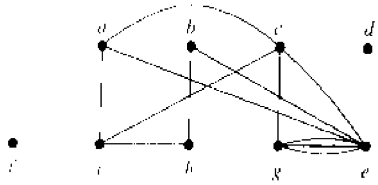
1.



2.



3.



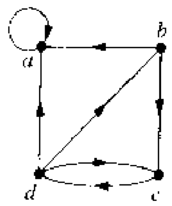
4. Find the sum of the degrees of the vertices of each graph in Exercises 1–3 and verify that it equals twice the number of edges in the graph.

5. Can a simple graph exist with 15 vertices each of degree 5?

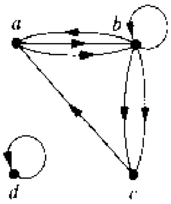
6. Show that the sum, over the set of people at a party, of the number of people a person has shaken hands with, is even. Assume that no one shakes his or her own hand.

In Exercises 7–9 determine the number of vertices and edges and find the in-degree and out-degree of each vertex for the given directed multigraph.

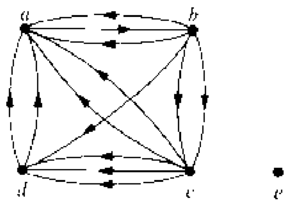
7.



8.



9.



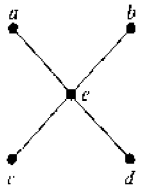
10. For each of the graphs in Exercises 7–9 determine the sum of the in-degrees of the vertices and the sum of the out-degrees of the vertices directly. Show that they are both equal to the number of edges in the graph.

11. Construct the underlying undirected graph for the graph with directed edges in Figure 2.

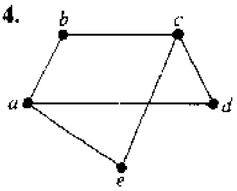
12. Draw the following graphs.
a) K_7 **b)** $K_{1,8}$ **c)** $K_{4,4}$
d) C_7 **e)** W_7 **f)** Q_4

In Exercises 13–17 determine whether the graph is bipartite.

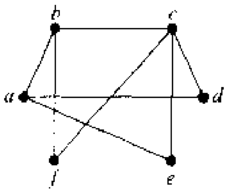
13.



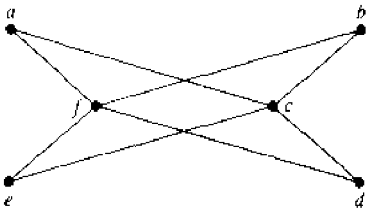
14.



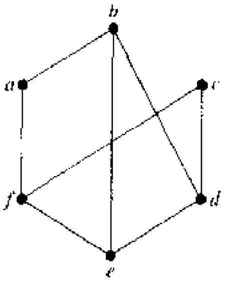
15.



16.



17.



18. For which values of n are the following graphs bipartite?
a) K_n **b)** C_n **c)** W_n **d)** Q_n

19. How many vertices and how many edges do the following graphs have?
a) K_n **b)** C_n **c)** W_n
d) $K_{m,n}$ **e)** Q_n

20. How many edges does a graph have if it has vertices of degree 4, 3, 3, 2, 2? Draw such a graph.

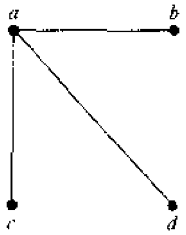
21. Does there exist a simple graph with five vertices of the following degrees? If so, draw such a graph.
a) 3, 3, 3, 3, 2 **b)** 1, 2, 3, 4, 5
c) 1, 2, 3, 4, 4 **d)** 3, 4, 3, 4, 3
e) 0, 1, 2, 2, 3 **f)** 1, 1, 1, 1, 1

22. How many subgraphs with at least one vertex does K_2 have?

23. How many subgraphs with at least one vertex does K_3 have?

24. How many subgraphs with at least one vertex does W_3 have?

25. Draw all subgraphs of the following graph.



26. Let G be a graph with v vertices and e edges. Let M be the maximum degree of the vertices of G, and let m be the minimum degree of the vertices of G. Show that
a) $2e/v \geq m$. **b)** $2e/v \leq M$.

A simple graph is called **regular** if every vertex of this graph has the same degree. A regular graph is called n-**regular** if every vertex in this graph has degree n.

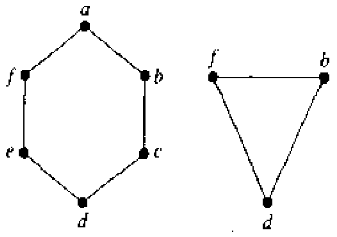
27. For which values of n are the following graphs regular?
a) K_n **b)** C_n **c)** W_n **d)** Q_n

28. For which values of m and n is $K_{m,n}$ regular?

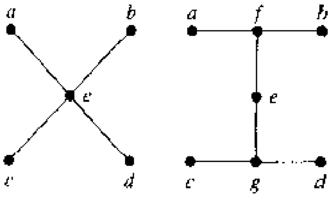
29. How many vertices does a regular graph of degree 4 with 10 edges have?

In Exercises 30–32 find the union of the given pair of simple graphs. (Assume edges with the same endpoints are the same.)

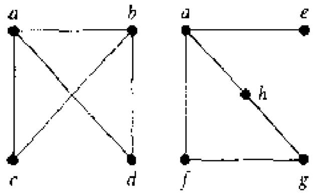
30.



31.



32.



33. The **complementary graph** $\overline{G}$ of a simple graph G has the same vertices as G. Two vertices are adjacent in $\overline{G}$ if and only if they are not adjacent in G. Find the following.

a) $\overline{K_n}$ b) $\overline{K_{m,n}}$
c) $\overline{C_n}$ d) $\overline{Q_n}$

34. If G is a simple graph with 15 edges and $\overline{G}$ has 13 edges, how many vertices does G have?

35. If the simple graph G has v vertices and e edges, how many edges does $\overline{G}$ have?

*36. Show that if G is a bipartite simple graph with v vertices and e edges, then $e \leq v^2/4$.

37. Show that if G is a simple graph with n vertices, then the union of G and $\overline{G}$ is K_n.

*38. Describe an algorithm to decide whether a graph is bipartite.

The **converse** of a directed graph $G = (V, E)$, denoted by G^c, is the directed graph (V, F) where $(u, v) \in F$ if and only if $(v, u) \in E$.

39. Draw the converse of each of the graphs in Exercises 7–9 in Section 7.1.

40. Show that $(G^c)^c = G$ whenever G is a directed graph.

41. Show that the graph G is its own converse if and only if the relation associated with G (see Section 6.3) is symmetric.

42. Extend the definition of the converse of a directed graph to the notion of the converse of a directed multigraph.

43. Draw the mesh network for interconnecting nine parallel processors.

44. In a variant of a mesh network for interconnecting $n = m^2$ processors, processor $P(i, j)$ is connected to the four processors $P((i \pm 1) \mathbf{mod}\ m, j)$, $P(i, (j \pm 1) \mathbf{mod}\ m)$, so that connections wrap around the edges of the mesh. Draw this variant of the mesh network for 16 processors.

45. Show that every pair of processors in a mesh network of $n = m^2$ processors can communicate using $O(\sqrt{n}) = O(m)$ hops between directly connected processors.

7.3

Representing Graphs and Graph Isomorphism

INTRODUCTION

There are many useful ways to represent graphs. As we will see throughout this chapter, in working with a graph it is helpful to be able to choose its most convenient representation. In this section we will show how to represent graphs in several different ways.

Sometimes, two graphs have exactly the same form, in the sense that there is a one-to-one correspondence between their vertex sets that preserves edges. In such a case, we say that the two graphs are **isomorphic.** Determining whether two graphs are isomorphic is an important problem of graph theory that we will study in this section.

REPRESENTING GRAPHS

One way to represent a graph without multiple edges is to list all the edges of this graph. Another way to represent a graph with no multiple edges is to use **adjacency lists,** which specify the vertices that are adjacent to each vertex of the graph.

EXAMPLE 1 Use adjacency lists to describe the simple graph given in Figure 1.

Solution: Table 1 lists those vertices adjacent to each of the vertices of the graph. ■

EXAMPLE 2 Represent the directed graph shown in Figure 2 by listing all the vertices that are the terminal vertices of edges starting at each vertex of the graph.

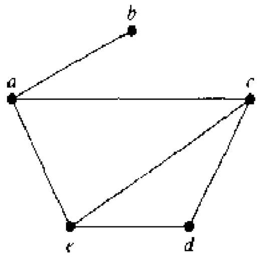


FIGURE 1 A Simple Graph.

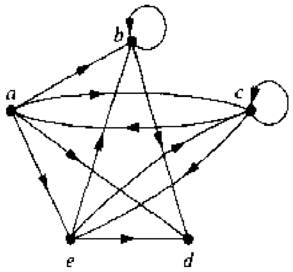


FIGURE 2 A Directed Graph.

TABLE 1 An Edge List for a Simple Graph.

Vertex	*Adjacent Vertices*
a	*b, c, e*
b	*a*
c	*a, d, e*
d	*c, e*
e	*a, c, d*

TABLE 2 An Edge List for a Directed Graph.

Initial Vertex	*Terminal Vertices*
a	*b, c, d, e*
b	*b, d*
c	*a, c, e*
d	
e	*b, c, d*

Solution: Table 2 represents the directed graph shown in Figure 2. ∎

ADJACENCY MATRICES

Carrying out graph algorithms using the representation of graphs by lists of edges, or by adjacency lists, can be cumbersome if there are many edges in the graph. To simplify computation, graphs can be represented using matrices. Two types of matrices commonly used to represent graphs will be presented here. One is based on the adjacency of vertices, and the other is based on incidence of vertices and edges.

Suppose that $G = (V, E)$ is a simple graph where $|V| = n$. Suppose that the vertices of G are listed arbitrarily as $v_1, v_2, \ldots, v_n$. The **adjacency matrix A** (or $\mathbf{A}_G$) of G, with respect to this listing of the vertices, is the $n \times n$ zero–one matrix with 1 as its (i, j)th entry when v_i and v_j are adjacent, and 0 as its (i, j)th entry when they are not adjacent. In other words, if its adjacency matrix is $\mathbf{A} = [a_{ij}]$, then

$$a_{ij} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \text{ is an edge of } G, \\ 0 & \text{otherwise.} \end{cases}$$

Note that an adjacency matrix of a graph is based on the ordering chosen for the vertices. Hence, there are as many as $n!$ different adjacency matrices for a graph with n vertices, since there are $n!$ different orderings of n vertices.

The adjacency matrix of a simple graph is symmetric, that is, $a_{ij} = a_{ji}$, since both of these entries are 1 when v_i and v_j are adjacent, and both are 0 otherwise. Furthermore, since a simple graph has no loops, each entry a_{ii}, $i = 1, 2, 3, \ldots, n$, is 0.

Note that when there are relatively few edges in a graph, the adjacency matrix is a sparse matrix, that is, a matrix with few nonzero entries. There are special techniques for representing, and computing with, such matrices. Also, it sometimes may be more efficient to work with lists of edges when representing and working with such graphs.

EXAMPLE 3 Use an adjacency matrix to represent the graph shown in Figure 3.

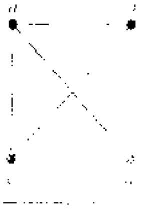


FIGURE 3 A Simple Graph.

Solution: We order the vertices as a, b, c, d. The matrix representing this graph is

$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

■

EXAMPLE 4 Draw a graph with the adjacency matrix

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

with respect to the ordering of vertices a, b, c, d.

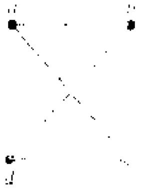


FIGURE 4 A Graph with the Given Adjacency Matrix.

Solution: A graph with this adjacency matrix is shown in Figure 4. ■

Adjacency matrices can also be used to represent undirected graphs with loops and with multiple edges. A loop at the vertex a_i is represented by a 1 at the (i, i)th position of the adjacency matrix. When multiple edges are present, the adjacency matrix is no longer a zero–one matrix, since the (i, j)th entry of this matrix equals the number of edges that are associated to $\{a_i, a_j\}$. All undirected graphs, including multigraphs and pseudographs, have symmetric adjacency matrices.

EXAMPLE 5 Use an adjacency matrix to represent the pseudograph shown in Figure 5.

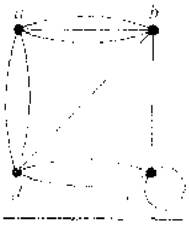


FIGURE 5 A Pseudograph.

Solution: The adjacency matrix using the ordering of vertices a, b, c, d is

$$\begin{bmatrix} 0 & 3 & 0 & 2 \\ 3 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \\ 2 & 1 & 2 & 0 \end{bmatrix}.$$

■

We used zero–one matrices in Chapter 6 to represent directed graphs. The matrix for a directed graph $G = (V, E)$ has a 1 in its (i, j)th position if there is an edge from v_i to v_j, where $v_1, v_2, \ldots, v_n$ is an arbitrary listing of the vertices of the directed graph. In other words, if $\mathbf{A} = [a_{ij}]$ is the adjacency matrix for the directed graph with respect to this listing of the vertices, then

$$a_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \text{ is an edge of } G, \\ 0 & \text{otherwise.} \end{cases}$$

The adjacency matrix for a directed graph does not have to be symmetric, since there may not be an edge from a_j to a_i when there is an edge from a_i to a_j.

Adjacency matrices can also be used to represent directed multigraphs. Again, such matrices are not zero–one matrices when there are multiple edges in the same direction connecting two vertices. In the adjacency matrix for a directed multigraph, a_{ij} equals the number of edges that are associated to (v_i, v_j).

INCIDENCE MATRICES

Another common way to represent graphs is to use **incidence matrices.** Let $G = (V, E)$ be an undirected graph. Suppose that $v_1, v_2, \ldots, v_n$ are the vertices and $e_1, e_2, \ldots, e_m$ are the edges of G. Then the incidence matrix with respect to this ordering of V and E is the $n \times m$ matrix $\mathbf{M} = [m_{ij}]$, where

$$m_{ij} = \begin{cases} 1 & \text{when edge } e_j \text{ is incident with } v_i, \\ 0 & \text{otherwise.} \end{cases}$$

EXAMPLE 6 Represent the graph shown in Figure 6 with an incidence matrix.

Solution: The incidence matrix is

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \left[\begin{array}{cccccc} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{array}\right] \end{array}.$$

■

Incidence matrices can also be used to represent multiple edges and loops. Multiple edges are represented in the incidence matrix using columns with identical entries, since these edges are incident with the same pair of vertices. Loops are represented using a column with exactly one entry equal to 1, corresponding to the vertex that is incident with this loop.

EXAMPLE 7 Represent the pseudograph shown in Figure 7 using an incidence matrix.

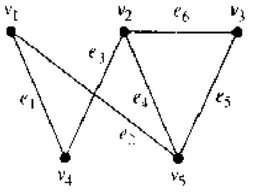


FIGURE 6 **An Undirected Graph.**

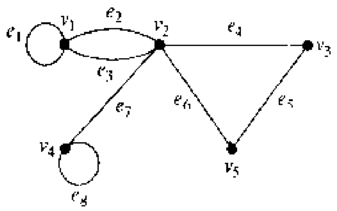


FIGURE 7 **A Pseudograph.**

Solution: The incidence matrix for this graph is

$$\begin{array}{c|cccccccc|} & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \\ v_1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ v_2 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ v_3 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ v_4 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ v_5 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{array}.$$

■

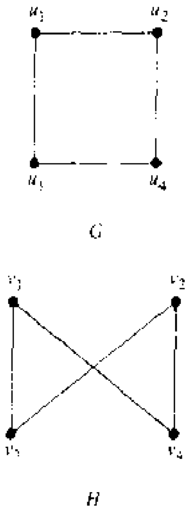


FIGURE 8 The Graphs *G* and *H*.

ISOMORPHISM OF GRAPHS

We often need to know whether it is possible to draw two graphs in the same way. For instance, in chemistry, graphs are used to model compounds. Different compounds can have the same molecular formula but can differ in structure. Such compounds will be represented by graphs that cannot be drawn in the same way. The graphs representing previously known compounds can be used to determine whether a supposedly new compound has been studied before.

There is a useful terminology for graphs with the same structure.

DEFINITION 1. The simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic* if there is a one-to-one and onto function f from V_1 to V_2 with the property that a and b are adjacent in G_1 if and only if $f(a)$ and $f(b)$ are adjacent in G_2, for all a and b in V_1. Such a function f is called an *isomorphism*.*

In other words, when two simple graphs are isomorphic, there is a one-to-one correspondence between vertices of the two graphs that preserves the adjacency relationship.

EXAMPLE 8 Show that the graphs $G = (V, E)$ and $H = (W, F)$, displayed in Figure 8, are isomorphic.

Solution: The function f with $f(u_1) = v_1$, $f(u_2) = v_4$, $f(u_3) = v_3$, and $f(u_4) = v_2$ is a one-to-one correspondence between V and W. To see that this correspondence preserves adjacency, note that adjacent vertices in G are u_1 and u_2, u_1 and u_3, u_2 and u_4, and u_3 and u_4, and each of the pairs $f(u_1) = v_1$ and $f(u_2) = v_4$, $f(u_1) = v_1$ and $f(u_3) = v_3$, $f(u_2) = v_4$ and $f(u_4) = v_2$, and $f(u_3) = v_3$ and $f(u_4) = v_2$ are adjacent in H. ■

web

It is often difficult to determine whether two simple graphs are isomorphic. There are $n!$ possible one-to-one correspondences between the vertex sets of two simple graphs with n vertices. Testing each such correspondence to see whether it preserves adjacency and nonadjacency is impractical if n is at all large.

However, we can often show that two simple graphs are not isomorphic by showing that they do not share a property that isomorphic simple graphs must both have. Such a property is called an **invariant** with respect to isomorphism of simple graphs. For

---

*The word *isomorphism* comes from the Greek roots *isos* for "equal" and *morphe* for "form."

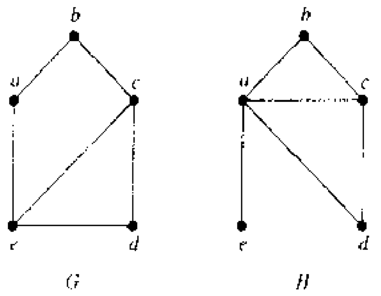


FIGURE 9 The Graphs *G* and *H*.

instance, isomorphic simple graphs must have the same number of vertices, since there is a one-to-one correspondence between the sets of vertices of the graphs. Furthermore, isomorphic simple graphs must have the same number of edges, because the one-to-one correspondence between vertices establishes a one-to-one correspondence between edges. In addition, the degrees of the vertices in isomorphic simple graphs must be the same. That is, a vertex v of degree d in G must correspond to a vertex $f(v)$ of degree d in H, since a vertex w in G is adjacent to v if and only if $f(v)$ and $f(w)$ are adjacent in H.

EXAMPLE 9 Show that the graphs displayed in Figure 9 are not isomorphic.

Solution: Both G and H have five vertices and six edges. However, H has a vertex of degree 1, namely, e, whereas G has no vertices of degree 1. It follows that G and H are not isomorphic. ■

The number of vertices, the number of edges, and the degrees of the vertices are all invariants under isomorphism. If any of these quantities differ in two simple graphs, these graphs cannot be isomorphic. However, when these invariants are the same, it does not necessarily mean that the two graphs are isomorphic. There are no useful sets of invariants currently known that can be used to determine whether simple graphs are isomorphic.

EXAMPLE 10 Determine whether the graphs shown in Figure 10 are isomorphic.

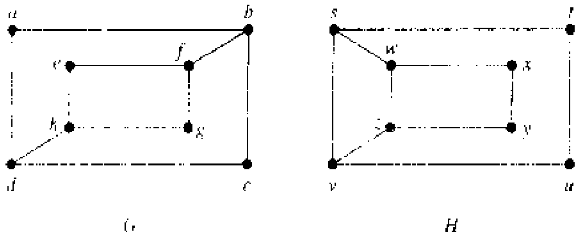


FIGURE 10 The Graphs *G* and *H*.

Solution: The graphs G and H both have eight vertices and 10 edges. They also both have four vertices of degree 2 and four of degree 3. Since these invariants all agree, it is still conceivable that these graphs are isomorphic.

However, G and H are not isomorphic. To see this, note that since $\deg(a) = 2$ in G, a must correspond to either t, u, x, or y in H, since these are the vertices of degree 2 in H. However, each of these four vertices in H is adjacent to another vertex of degree 2 in H, which is not true for a in G.

Another way to see that G and H are not isomorphic is to note that the subgraphs of G and H made up of vertices of degree 3 and the edges connecting them must be isomorphic if these two graphs are isomorphic (the reader should verify this). However, these subgraphs, shown in Figure 11, are not isomorphic. ■

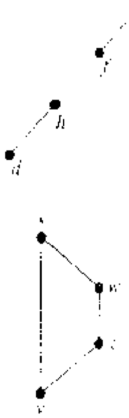


FIGURE 11 The Subgraphs of G and H Made Up of Vertices of Degree 3 and the Edges Connecting Them.

To show that a function f from the vertex set of a graph G to the vertex set of a graph H is an isomorphism, we need to show that f preserves edges. One helpful way to do this is to use adjacency matrices. In particular, to show that f is an isomorphism, we can show that the adjacency matrix of G is the same as the adjacency matrix of H, when rows and columns are labeled to correspond to the images under f of the vertices in G that are the labels of these rows and columns in the adjacency matrix of G. We illustrate how this is done in the following example.

EXAMPLE 11 Determine whether the graphs G and H displayed in Figure 12 are isomorphic.

Solution: Both G and H have six vertices and seven edges. Both have four vertices of degree 2 and two vertices of degree 3. It is also easy to see that the subgraphs of G and H consisting of all vertices of degree 2 and the edges connecting them are isomorphic (as the reader should verify). Since G and H agree with respect to these invariants, it is reasonable to try to find an isomorphism f.

We now will define a function f and then determine whether it is an isomorphism. Since $\deg(u_1) = 2$ and since u_1 is not adjacent to any other vertex of degree 2, the image of u_1 must be either v_4 or v_6, the only vertices of degree 2 in H not adjacent to a vertex of degree 2. We arbitrarily set $f(u_1) = v_6$. [If we found that this choice did not lead to isomorphism, we would then try $f(u_1) = v_4$.] Since u_2 is adjacent to u_1, the possible images of u_2 are v_3 and v_5. We arbitrarily set $f(u_2) = v_3$. Continuing in this way, using adjacency of vertices and degrees as a guide, we set $f(u_3) = v_4$, $f(u_4) = v_5$, $f(u_5) = v_1$, and $f(u_6) = v_2$. We now have a one-to-one correspondence between the vertex set of G and the vertex set of H, namely: $f(u_1) = v_6$, $f(u_2) = v_3$, $f(u_3) = v_4$, $f(u_4) = v_5$, $f(u_5) = v_1$, $f(u_6) = v_2$. To see whether f preserves edges,

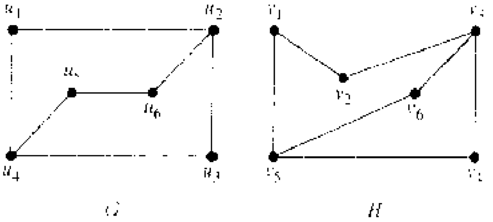


FIGURE 12 Graphs G and H.

we examine the adjacency matrix of G,

$$\mathbf{A}_G = \begin{array}{c|cccccc} & u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \\ \hline u_1 & 0 & 1 & 0 & 1 & 0 & 0 \\ u_2 & 1 & 0 & 1 & 0 & 0 & 1 \\ u_3 & 0 & 1 & 0 & 1 & 0 & 0 \\ u_4 & 1 & 0 & 1 & 0 & 1 & 0 \\ u_5 & 0 & 0 & 0 & 1 & 0 & 1 \\ u_6 & 0 & 1 & 0 & 0 & 1 & 0 \end{array},$$

and the adjacency matrix of H with the rows and columns labeled by the images of the corresponding vertices in G,

$$\mathbf{A}_H = \begin{array}{c|cccccc} & v_6 & v_3 & v_4 & v_5 & v_1 & v_2 \\ \hline v_6 & 0 & 1 & 0 & 1 & 0 & 0 \\ v_3 & 1 & 0 & 1 & 0 & 0 & 1 \\ v_4 & 0 & 1 & 0 & 1 & 0 & 0 \\ v_5 & 1 & 0 & 1 & 0 & 1 & 0 \\ v_1 & 0 & 0 & 0 & 1 & 0 & 1 \\ v_2 & 0 & 1 & 0 & 0 & 1 & 0 \end{array}.$$

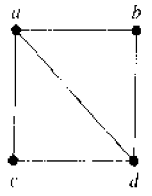
Since $\mathbf{A}_G = \mathbf{A}_H$, it follows that f preserves edges. We conclude that f is an isomorphism, so that G and H are isomorphic. Note that if f turned out not to be an isomorphism, we would *not* have established that G and H are not isomorphic, since another correspondence of the vertices in G and H may be an isomorphism. ■

web The best algorithms known for determining whether two graphs are isomorphic have exponential worst-case time complexity (in the number of vertices of the graphs). However, linear average-case time complexity algorithms are known that solve this problem, and there is some hope that an algorithm with polynomial worst-case time complexity for determining whether two graphs are isomorphic can be found. The best practical algorithm, called NAUTY, can be used to determine whether two graphs with as many as 100 vertices are isomorphic in less than 1 second on a modern PC. The software for NAUTY can be downloaded over the Internet and experimented with.

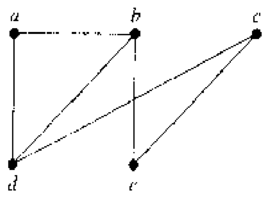
Exercises

In Exercises 1–4 use an adjacency list to represent the given graph.

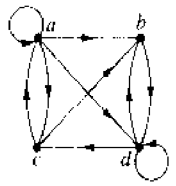
1.



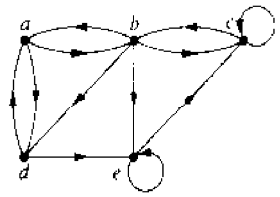
2.



3.



4.



5. Represent the graph in Exercise 1 with an adjacency matrix.

6. Represent the graph in Exercise 2 with an adjacency matrix.
7. Represent the graph in Exercise 3 with an adjacency matrix.
8. Represent the graph in Exercise 4 with an adjacency matrix.
9. Represent each of the following graphs with an adjacency matrix.
   a) K_4 b) $K_{1,4}$
   c) $K_{2,3}$ d) C_4
   e) W_4 f) Q_3

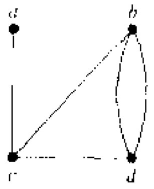
In Exercises 10–12 draw a graph with the given adjacency matrix.

10. $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

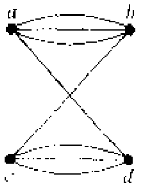
11. $\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$ **12.** $\begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix}$

In Exercises 13–15 represent the given graph using an adjacency matrix.

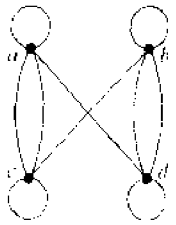
13.



14.



15.



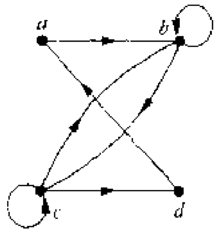
In Exercises 16–18 draw an undirected graph represented by the given adjacency matrix.

16. $\begin{bmatrix} 1 & 3 & 2 \\ 3 & 0 & 4 \\ 2 & 4 & 0 \end{bmatrix}$

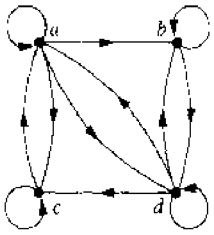
17. $\begin{bmatrix} 1 & 2 & 0 & 1 \\ 2 & 0 & 3 & 0 \\ 0 & 3 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$ **18.** $\begin{bmatrix} 0 & 1 & 3 & 0 & 4 \\ 1 & 2 & 1 & 3 & 0 \\ 3 & 1 & 1 & 0 & 1 \\ 0 & 3 & 0 & 0 & 2 \\ 4 & 0 & 1 & 2 & 3 \end{bmatrix}$

In Exercises 19–21 find the adjacency matrix of the given directed multigraph.

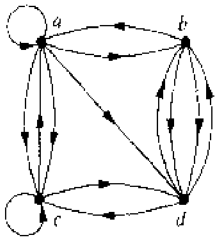
19.



20.



21.



In Exercises 22–24 draw the graph represented by the given adjacency matrix.

22. $\begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ **23.** $\begin{bmatrix} 1 & 2 & 1 \\ 2 & 0 & 0 \\ 0 & 2 & 2 \end{bmatrix}$

24. $\begin{bmatrix} 0 & 2 & 3 & 0 \\ 1 & 2 & 2 & 1 \\ 2 & 1 & 1 & 0 \\ 1 & 0 & 0 & 2 \end{bmatrix}$

25. Is every zero–one square matrix that is symmetric and has zeros on the diagonal the adjacency matrix of a simple graph?

26. Use an incidence matrix to represent the graphs in Exercises 1 and 2.

27. Use an incidence matrix to represent the graphs in Exercises 13–15.

***28.** What is the sum of the entries in a row of the adjacency matrix for an undirected graph? For a directed graph?

***29.** What is the sum of the entries in a column of the adjacency matrix for an undirected graph? For a directed graph?

30. What is the sum of the entries in a row of the incidence matrix for an undirected graph?

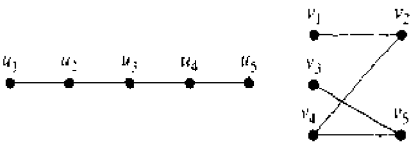
31. What is the sum of the entries in a column of the incidence matrix for an undirected graph?

***32.** Find an adjacency matrix for each of the following.
a) K_n **b)** C_n **c)** W_n **d)** $K_{m,n}$ **e)** Q_n

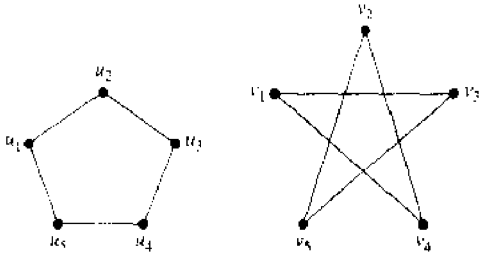
***33.** Find incidence matrices for the graphs in parts (a)–(d) of Exercise 32.

In Exercises 34–44 determine whether the given pair of graphs is isomorphic.

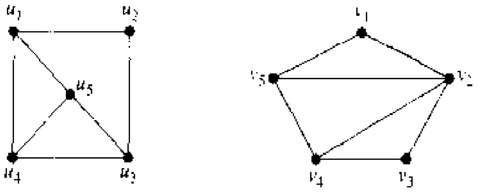
34.


u_1
u_2
u_3
u_4
u_5
v_1
v_2
v_3
v_4
v_5

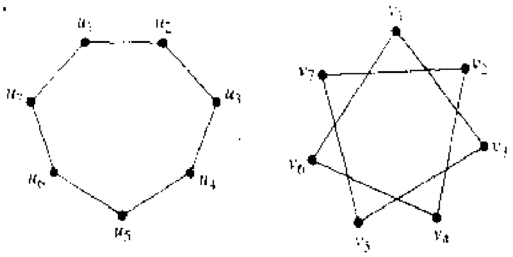
35.


u_2
u_1
u_3
u_5
u_4
v_2
v_1
v_3
v_5
v_4

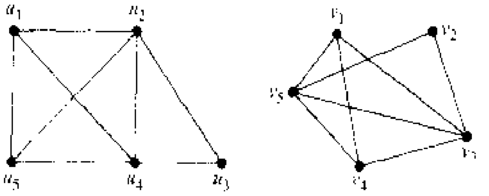
36.


u_1
u_2
u_5
u_4
u_3
v_1
v_5
v_2
v_4
v_3

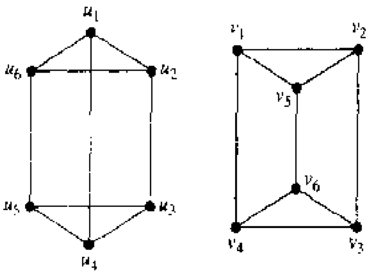
37.


u_1
u_2
u_7
u_3
u_6
u_4
u_5
v_1
v_7
v_2
v_6
v_3
v_5
v_4

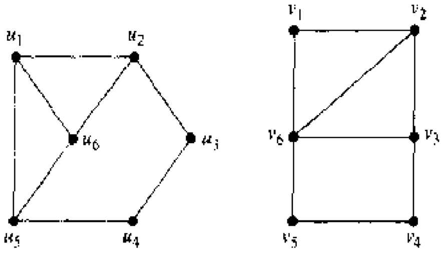
38.


u_1
u_2
u_5
u_4
u_3
v_1
v_2
v_5
v_3
v_4

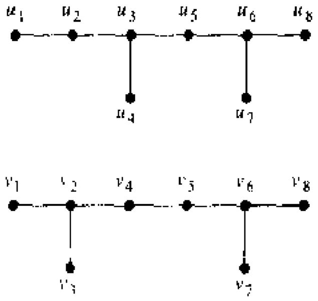
39.


u_1
u_6
u_2
u_5
u_3
u_4
v_1
v_2
v_5
v_6
v_4
v_3

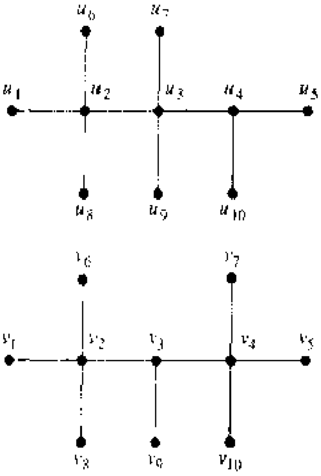
40.


u_1
u_2
u_6
u_3
u_5
u_4
v_1
v_2
v_6
v_3
v_5
v_4

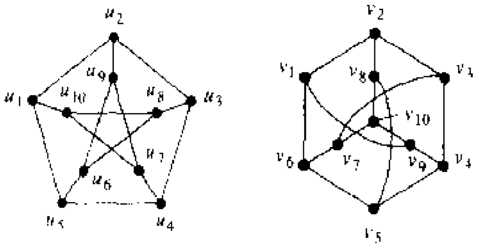
41.


u_1
u_2
u_3
u_5
u_6
u_8
u_4
u_7
v_1
v_2
v_4
v_5
v_6
v_8
v_3
v_7

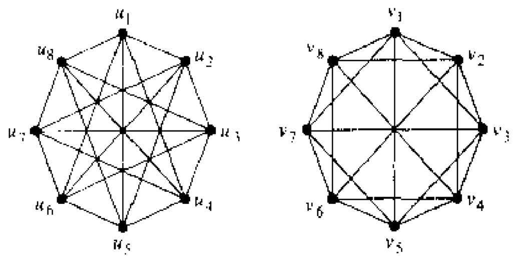
42.


u_6
u_7
u_1
u_2
u_3
u_4
u_5
u_8
u_9
u_{10}
v_6
v_7
v_1
v_2
v_3
v_4
v_5
v_8
v_9
v_{10}

43.


u_2
u_9
u_1
u_{10}
u_8
u_3
u_7
u_6
u_5
u_4
v_2
v_1
v_8
v_3
v_{10}
v_6
v_7
v_9
v_4
v_5

44.



45. Show that isomorphism of simple graphs is an equivalence relation.

46. Suppose that G and H are isomorphic simple graphs. Show that their complementary graphs $\overline{G}$ and $\overline{H}$ are also isomorphic.

47. Describe the row and column of an adjacency matrix of a graph corresponding to an isolated vertex.

48. Describe the row of an incidence matrix of a graph corresponding to an isolated vertex.

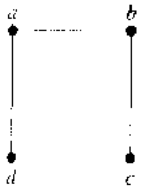
49. Show that the vertices of a bipartite graph with two or more vertices can be ordered so that its adjacency matrix has the form

$$\begin{bmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{B} & \mathbf{0} \end{bmatrix}$$

where the four entries shown are rectangular blocks.

A simple graph G is called **self-complementary** if G and $\overline{G}$ are isomorphic.

50. Show that the following graph is self-complementary.



51. Find a self-complementary simple graph with five vertices.

*52. Show that if G is a self-complementary simple graph with v vertices, then $v \equiv 0$ or $1 \pmod 4$.

53. For which integers n is C_n self-complementary?

54. How many nonisomorphic simple graphs are there with n vertices, when n is

a) 2? b) 3? c) 4?

55. How many nonisomorphic simple graphs are there with five vertices and three edges?

56. How many nonisomorphic simple graphs are there with six vertices and four edges?

57. Are the simple graphs with the following adjacency matrices isomorphic?

a) $\begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$

b) $\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$

c) $\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$

58. Determine whether the graphs without loops with the following incidence matrices are isomorphic.

a) $\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$

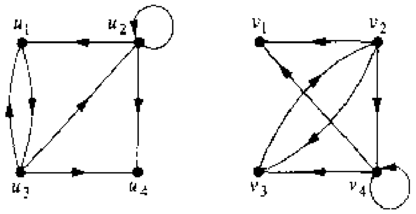
b) $\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$

59. Extend the definition of isomorphism of simple graphs to undirected graphs containing loops and multiple edges.

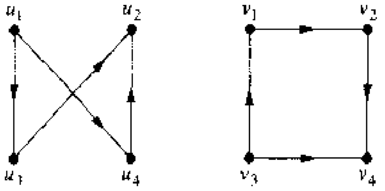
60. Define isomorphism of directed graphs.

In Exercises 61–64 determine whether the given pair of directed graphs is isomorphic.

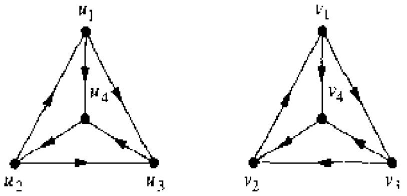
61.



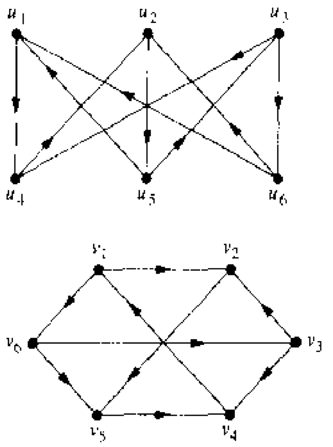
62.



63.



64.



65. Show that if G and H are isomorphic directed graphs, then the converses of G and H (defined in the preamble of Exercise 39 of Section 7.2) are also isomorphic.

***66.** How many nonisomorphic directed simple graphs are there with n vertices, when n is
- **a)** 2?
- **b)** 3?
- **c)** 4?

***67.** What is the product of the incidence matrix and its transpose for an undirected graph?

***68.** How much storage is needed to represent a simple graph with v vertices and e edges using
- **a)** adjacency lists?
- **b)** an adjacency matrix?
- **c)** an incidence matrix?

A **devil's pair** for a purported isomorphism test is a pair of nonisomorphic graphs that the test fails to show are not isomorphic.

69. Find a devil's pair for the test that checks the sequence of degrees of the vertices in the two graphs to make sure they agree.

7.4

Connectivity

INTRODUCTION

Many problems can be modeled with paths formed by traveling along the edges of graphs. For instance, the problem of determining whether a message can be sent between two computers using intermediate links can be studied with a graph model. Problems of efficiently planning routes for mail delivery, garbage pickup, diagnostics in computer networks, and so on, can be solved using models that involve paths in graphs.

PATHS

We begin by defining the basic terminology of graph theory that deals with paths.

DEFINITION 1. A *path* of *length* n from u to v, where n is a positive integer, in an undirected graph is a sequence of edges $e_1, \ldots, e_n$ of the graph such that $f(e_1) = \{x_0, x_1\}, f(e_2) = \{x_1, x_2\}, \ldots, f(e_n) = \{x_{n-1}, x_n\}$, where $x_0 = u$ and $x_n = v$. When the graph is simple, we denote this path by its vertex sequence $x_0, x_1, \ldots, x_n$ (since listing these vertices uniquely determines the path). The path is a *circuit* if it begins and ends at the same vertex, that is, if $u = v$. The path or circuit is said to *pass through* or *traverse* the vertices $x_1, x_2, \ldots, x_{n-1}$. A path or circuit is *simple* if it does not contain the same edge more than once.

When it is not necessary to distinguish between multiple edges, we will denote a path $e_1, e_2, \ldots, e_n$ where $f(e_i) = \{x_{i-1}, x_i\}$ for $i = 1, 2, \ldots, n$ by its vertex sequence $x_0, x_1, \ldots, x_n$. This notation identifies a path only up to the vertices it passes through. There may be more than one path that passes through this sequence of vertices.

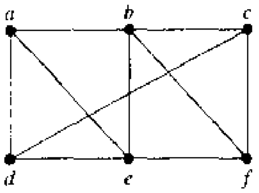


FIGURE 1 A Simple Graph.

EXAMPLE 1 In the simple graph shown in Figure 1, a, d, c, f, e is a simple path of length 4, since $\{a, d\}$, $\{d, c\}$, $\{c, f\}$, and $\{f, e\}$ are all edges. However, d, e, c, a is not a path, since $\{e, c\}$ is not an edge. Note that b, c, f, e, b is a circuit of length 4 since $\{b, c\}$, $\{c, f\}$, $\{f, e\}$, and $\{e, b\}$ are edges, and this path begins and ends at b. The path a, b, e, d, a, b, which is of length 5, is not simple since it contains the edge $\{a, b\}$ twice. ■

Paths and circuits in directed graphs were introduced in Chapter 6. We now define such paths for directed multigraphs.

DEFINITION 2. A *path* of length n, where n is a positive integer, from u to v in a directed multigraph is a sequence of edges $e_1, e_2, \ldots, e_n$ of the graph such that $f(e_1) = (x_0, x_1)$, $f(e_2) = (x_1, x_2), \ldots, f(e_n) = (x_{n-1}, x_n)$, where $x_0 = u$ and $x_n = v$. When there are no multiple edges in the graph, this path is denoted by its vertex sequence $x_0, x_1, x_2, \ldots, x_n$. A path that begins and ends at the same vertex is called a *circuit* or *cycle*. A path or circuit is called *simple* if it does not contain the same edge more than once.

When it is not necessary to distinguish between multiple edges, we will denote a path $e_1, e_2, \ldots, e_n$ where $f(e_i) = (x_{i-1}, x_i)$ for $i = 1, 2, \ldots, n$ by its vertex sequence $x_0, x_1, \ldots, x_n$. The notation identifies a path only up to the vertices it passes through. There may be more than one path that passes through this sequence of vertices.

CONNECTEDNESS IN UNDIRECTED GRAPHS

When does a computer network have the property that every pair of computers can share information, if messages can be sent through one or more intermediate computers? When a graph is used to represent this computer network, where vertices represent the computers and edges represent the communications links, this question becomes: When is there always a path between two vertices in the graph?

DEFINITION 3. An undirected graph is called *connected* if there is a path between every pair of distinct vertices of the graph.

Thus, any two computers in the network can communicate if and only if the graph of this network is connected.

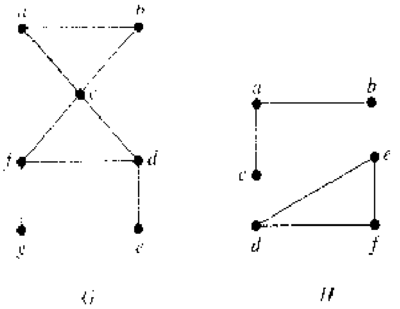


FIGURE 2 The Graphs *G* and *H*.

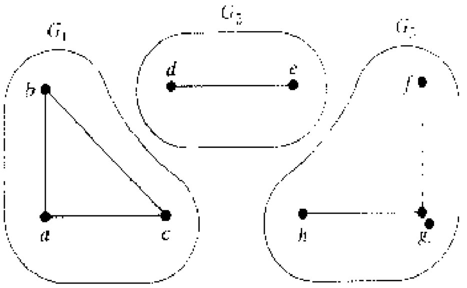


FIGURE 3 The Graph *G* and Its Connected Components G_1, G_2, and G_3.

EXAMPLE 2 The graph G in Figure 2 is connected, since for every pair of distinct vertices there is a path between them (the reader should verify this). However, the graph H in Figure 2 is not connected. For instance, there is no path in H between vertices a and d. ■

We will need the following theorem in Chapter 8.

THEOREM 1 **There is a simple path between every pair of distinct vertices of a connected undirected graph.**

Proof: Let u and v be two distinct vertices of the connected undirected graph $G = (V, E)$. Since G is connected, there is at least one path between u and v. Let $x_0, x_1, \ldots, x_n$, where $x_0 = u$ and $x_n = v$, be the vertex sequence of a path of least length. This path of least length is simple. To see this, suppose it is not simple. Then $x_i = x_j$ for some i and j with $0 \leq i < j$. This means that there is a path from u to v of shorter length with vertex sequence $x_0, x_1, \ldots, x_{i-1}, x_j, \ldots, x_n$ obtained by deleting the edges corresponding to the vertex sequence $x_i, \ldots, x_{j-1}$. □

A graph that is not connected is the union of two or more connected subgraphs, each pair of which has no vertex in common. These disjoint connected subgraphs are called the **connected components** of the graph.

EXAMPLE 3 What are the connected components of the graph G shown in Figure 3?

Solution: The graph G is the union of three disjoint connected subgraphs G_1, G_2, and G_3, shown in Figure 3. These three subgraphs are the connected components of G. ■

Sometimes the removal of a vertex and all edges incident with it produces a subgraph with more connected components than in the original graph. Such vertices are called **cut vertices** (or **articulation points**). The removal of a cut vertex from a connected graph produces a subgraph that is not connected. Analogously, an edge whose removal produces a graph with more connected components than in the original graph is called a **cut edge** or **bridge.**

EXAMPLE 4 Find the cut vertices and cut edges in the graph G shown in Figure 4.

Solution: The cut vertices of G are b, c, and e. The removal of one of these vertices (and its adjacent edges) disconnects the graph. The cut edges are $\{a, b\}$ and $\{c, e\}$. Removing either one of these edges disconnects G. ■

CONNECTEDNESS IN DIRECTED GRAPHS

There are two notions of connectedness in directed graphs, depending on whether the directions of the edges are considered.

DEFINITION 4. **A directed graph is *strongly connected* if there is a path from a to b and from b to a whenever a and b are vertices in the graph.**

For a directed graph to be strongly connected there must be a sequence of directed edges from any vertex in the graph to any other vertex. A directed graph can fail to be strongly connected but still be in "one piece." To make this precise, the following definition is given.

DEFINITION 5. **A directed graph is *weakly connected* if there is a path between any two vertices in the underlying undirected graph.**

That is, a directed graph is weakly connected if and only if there is always a path between two vertices when the directions of the edges are disregarded. Clearly, any strongly connected directed graph is also weakly connected.

EXAMPLE 5 Are the directed graphs G and H shown in Figure 5 strongly connected? Are they weakly connected?

Solution: G is strongly connected because there is a path between any two vertices in this directed graph (the reader should verify this). Hence, G is also weakly connected. The graph H is not strongly connected. There is no directed path from a to b in this graph. However, H is weakly connected, since there is a path between any two vertices in the underlying undirected graph of H (the reader should verify this). ■

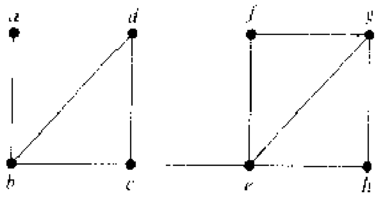


FIGURE 4 The Graph G.

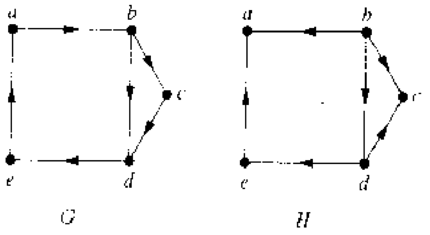


FIGURE 5 The Directed Graphs G and H.

PATHS AND ISOMORPHISM

There are several ways that paths and circuits can help determine whether two graphs are isomorphic. For example, the existence of a simple circuit of a particular length is a useful invariant that can be used to show that two graphs are not isomorphic. In addition, paths can be used to construct mappings that may be isomorphisms.

As we mentioned, a useful isomorphic invariant for simple graphs is the existence of a simple circuit of length k, where k is a positive integer greater than 2. (The proof that this is an invariant is left as Exercise 36 at the end of this section.) Example 6 illustrates how this invariant can be used to show that two graphs are not isomorphic.

EXAMPLE 6 Determine whether the graphs G and H shown in Figure 6 are isomorphic.

Solution: Both G and H have six vertices and eight edges. Each has four vertices of degree 3, and two vertices of degree 2. So, the three invariants—number of vertices, number of edges, and degrees of vertices—all agree for the two graphs. However, H has a simple circuit of length 3, namely, v_1, v_2, v_6, v_1 whereas G has no simple circuit of length 3, as can be determined by inspection (all simple circuits in G have length at least 4). Since the existence of a simple circuit of length 3 is an isomorphic invariant, G and H are not isomorphic. ■

We have shown how the existence of a type of path, namely, a simple circuit of a particular length, can be used to show that two graphs are not isomorphic. We can also use paths to find mappings that are potential isomorphisms.

EXAMPLE 7 Determine whether the graphs G and H shown in Figure 7 are isomorphic.

Solution: Both G and H have five vertices and six edges, both have two vertices of degree 3 and three vertices of degree 2, and both have a simple circuit of length 3, a simple circuit of length 4, and a simple circuit of length 5. Since all these isomorphic invariants agree, G and H may be isomorphic. To find a possible isomorphism, we can follow paths that go through all vertices so that the corresponding vertices in the two graphs have the same degree. For example, the paths u_1, u_4, u_3, u_2, u_5 in G and $v_3, v_2,$

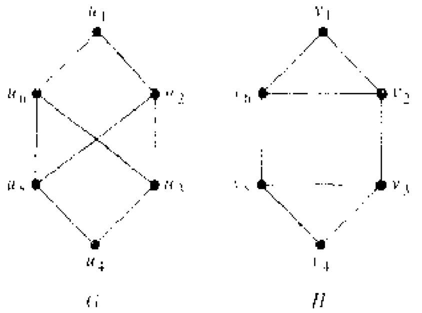


FIGURE 6 The Graphs G and H.

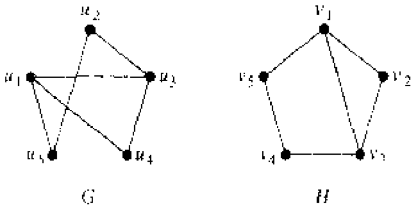


FIGURE 7 The Graphs G and H.

v_1, v_5, v_4 in H both go through every vertex in the graph, start at a vertex of degree 3, go through vertices of degrees 2, 3, and 2, respectively, and end at a vertex of degree 2. By following these paths through the graphs, we define the mapping f with $f(u_1) = v_3$, $f(u_4) = v_2$, $f(u_3) = v_1$, $f(u_2) = v_5$, and $f(u_5) = v_4$. The reader can show that f is an isomorphism, so that G and H are isomorphic, either by showing that f preserves edges or by showing that with the appropriate orderings of vertices the adjacency matrices of G and H are the same. ■

COUNTING PATHS BETWEEN VERTICES

The number of paths between two vertices in a graph can be determined using its adjacency matrix.

THEOREM 2

Let G be a graph with adjacency matrix $\mathbf{A}$ with respect to the ordering $v_1, v_2, \ldots, v_n$ (with directed or undirected edges, with multiple edges and loops allowed). The number of different paths of length r from v_i to v_j, where r is a positive integer, equals the (i, j)th entry of $\mathbf{A}^r$.

Proof: The theorem will be proved using mathematical induction. Let G be a graph with adjacency matrix $\mathbf{A}$ (assuming an ordering $v_1, v_2, \ldots, v_n$ of the vertices of G). The number of paths from v_i to v_j of length 1 is the (i, j)th entry of $\mathbf{A}$, since this entry is the number of edges from v_i to v_j.

Assume that the (i, j)th entry of $\mathbf{A}^r$ is the number of different paths of length r from v_i to v_j. This is the induction hypothesis. Since $\mathbf{A}^{r+1} = \mathbf{A}^r\mathbf{A}$, the (i, j)th entry of $\mathbf{A}^{r+1}$ equals

$$b_{i1}a_{1j} + b_{i2}a_{2j} + \cdots + b_{in}a_{nj}$$

where b_{ik} is the (i, k)th entry of $\mathbf{A}^r$. By the induction hypothesis, b_{ik} is the number of paths of length r from v_i to v_k.

A path of length $r + 1$ from v_i to v_j is made up of a path of length r from v_i to some intermediate vertex v_k, and an edge from v_k to v_j. By the product rule for counting, the number of such paths is the product of the number of paths of length r from v_i to v_k, namely, b_{ik}, and the number of edges from v_k to v_j, namely, a_{kj}. When these products are added for all possible intermediate vertices v_k, the desired result follows by the sum rule for counting. □

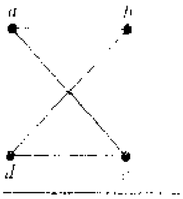


FIGURE 8 The Graph G.

EXAMPLE 8

How many paths of length 4 are there from a to d in the simple graph G shown in Figure 8?

Solution: The adjacency matrix of G (ordering the vertices as a, b, c, d) is

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

Hence, the number of paths of length 4 from a to d is the $(1, 4)$th entry of $\mathbf{A}^4$. Since

$$\mathbf{A}^4 = \begin{bmatrix} 8 & 0 & 0 & 8 \\ 0 & 8 & 8 & 0 \\ 0 & 8 & 8 & 0 \\ 8 & 0 & 0 & 8 \end{bmatrix},$$

there are exactly eight paths of length 4 from a to d. By inspection of the graph, we see that a, b, a, b, d; a, b, a, c, d; a, b, d, b, d; a, b, d, c, d; a, c, a, b, d; a, c, a, c, d; a, c, d, b, d; and a, c, d, c, d are the eight paths from a to d. ■

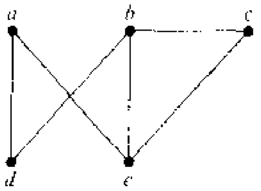
Theorem 2 can be used to find the length of the shortest path between two vertices of a graph (see Exercise 32), and it can also be used to determine whether a graph is connected (see Exercises 37 and 38).

Exercises

1. Does each of the following lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?

a) a, e, b, c, b **b)** a, e, a, d, b, c, a

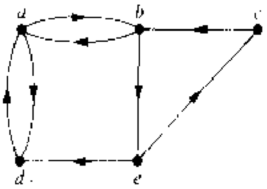
c) e, b, a, d, b, e **d)** c, b, d, a, e, c



2. Does each of the following lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?

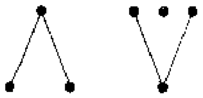
a) a, b, e, c, b **b)** a, d, a, d, a

c) a, d, b, e, a **d)** a, b, e, c, b, d, a

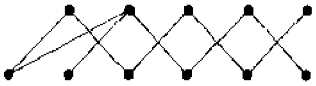


In Exercises 3–5 determine whether the given graph is connected.

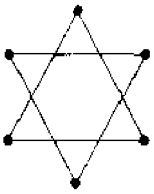
3.



4.



5.



6. How many connected components does each of the graphs in Exercises 3–5 have? For each graph find each of its connected components.

***7.** Find the number of paths of length n between two different vertices in K_4 if n is

a) 2. **b)** 3. **c)** 4. **d)** 5.

***8.** Find the number of paths of length n between any two adjacent vertices in $K_{3,3}$ for the values of n in Exercise 7.

***9.** Find the number of paths of length n between any two nonadjacent vertices in $K_{3,3}$ for the values of n in Exercise 7.

10. Find the number of paths between c and d in the graph in Figure 1 of length
a) 2. b) 3. c) 4. d) 5. e) 6. f) 7.

11. Find the number of paths from a to e in the directed graph in Exercise 2 of length
a) 2. b) 3. c) 4. d) 5. e) 6. f) 7.

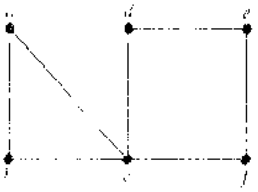
***12.** Show that a connected graph with n vertices has at least $n - 1$ edges.

13. Let $G = (V, E)$ be a simple graph. Let R be the relation on V consisting of pairs of vertices (u, v) such that there is a path from u to v or such that $u = v$. Show that R is an equivalence relation.

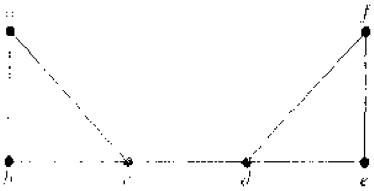
***14.** Show that in any simple graph there is a path from any vertex of odd degree to some other vertex of odd degree.

In Exercises 15–17 find all the cut vertices of the given graph.

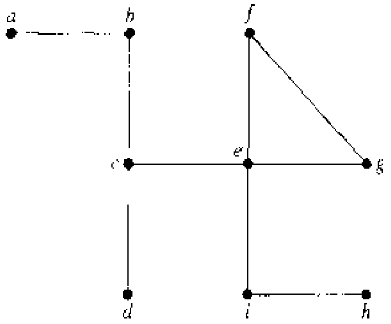
15.



16.



17.



18. Find all the cut edges in the graphs in Exercises 15–17.

***19.** Suppose that v is an endpoint of a cut edge. Prove that v is a cut vertex if and only if this vertex is not pendant.

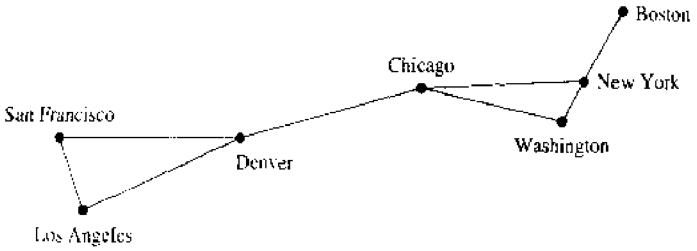
***20.** Show that a vertex c in the connected simple graph G is a cut vertex if and only if there are vertices u and v, both different from c, such that every path between u and v passes through c.

***21.** Show that a simple graph with at least two vertices has at least two vertices that are not cut vertices.

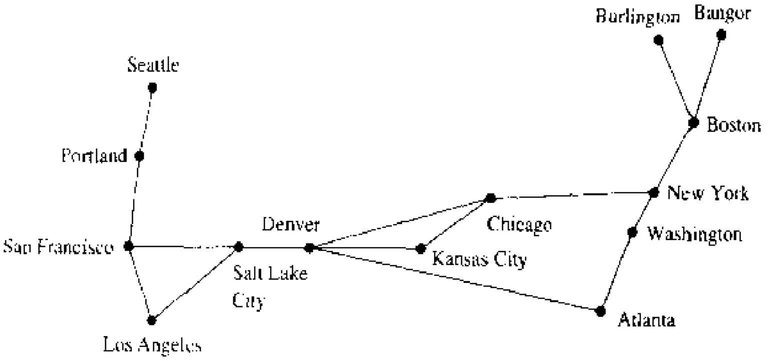
***22.** Show that an edge in a simple graph is a cut edge if and only if this edge is not part of any simple circuit in the graph.

23. A communications link in a network should be provided with a backup link if its failure makes it impossible for some message to be sent. For each of the communications networks shown below in (a) and (b), determine those links that should be backed up.

a)


Boston
Chicago
New York
San Francisco
Washington
Denver
Los Angeles

b)


Burlington
Bangor
Seattle
Boston
Portland
New York
Denver
Chicago
Washington
San Francisco
Kansas City
Salt Lake City
Atlanta
Los Angeles

A **vertex basis** *in a directed graph is a set of vertices such that there is a path to every vertex in the directed graph not in the set from some vertex in this set and there is no path from any vertex in the set to another vertex in the set.*

24. Find a vertex basis for each of the directed graphs in Exercises 7–9 of Section 7.2.

25. What is the significance of a vertex basis in an influence graph (described in Example 2 of Section 7.1)? Find a vertex basis in the influence graph in this example.

26. Show that if a connected simple graph G is the union of the graphs G_1 and G_2, then G_1 and G_2 have at least one common vertex.

***27.** Show that if a simple graph G has k connected components and these components have $n_1, n_2, \ldots, n_k$ vertices, respectively, then the number of edges of G does not exceed

$$\sum_{i=1}^{k} C(n_i, 2).$$

***28.** Use Exercise 27 to show that a simple graph with n vertices and k connected components has at most $(n-k)(n-k+1)/2$ edges. [*Hint:* First show that

$$\sum_{i=1}^{k} n_i^2 \le n^2 - (k-1)(2n-k),$$

where n_i is the number of vertices in the ith connected component.]

***29.** Show that a simple graph G with n vertices is connected if it has more than $(n-1)(n-2)/2$ edges.

30. Describe the adjacency matrix of a graph with n connected components when the vertices of the graph are listed so that vertices in each connected component are listed successively.

31. How many nonisomorphic connected simple graphs are there with n vertices when n is

a) 2? **b)** 3? **c)** 4? **d)** 5?

32. Explain how Theorem 2 can be used to find the length of the shortest path from a vertex v to a vertex w in a graph.

33. Use Theorem 2 to find the length of the shortest path between a and f in the multigraph in Figure 1.

34. Use Theorem 2 to find the length of the shortest path from a to c in the directed graph in Exercise 2.

35. Let P_1 and P_2 be two simple paths between the vertices u and v in the simple graph G that do not contain the same set of edges. Show that there is a simple circuit in G.

36. Show that the existence of a simple circuit of length k, where k is a positive integer greater than 2, is an isomorphic invariant.

37. Explain how Theorem 2 can be used to determine whether a graph is connected.

38. Use Exercise 37 to show that the graph G in Figure 2 is connected whereas the graph H in that figure is not connected.

7.5

Euler and Hamilton Paths

INTRODUCTION

web

The town of Königsberg, Prussia (now called Kaliningrad and part of the Russian republic), was divided into four sections by the branches of the Pregel River. These four sections included the two regions on the banks of the Pregel, Kneiphof Island, and the region between the two branches of the Pregel. In the eighteenth century seven bridges connected these regions. Figure 1 depicts these regions and bridges.

The townspeople took long walks through town on Sundays. They wondered whether it was possible to start at some location in the town, travel across all the bridges without crossing any bridge twice, and return to the starting point.

The Swiss mathematician Leonhard Euler solved this problem. His solution, published in 1736, may be the first use of graph theory. Euler studied this problem using the multigraph obtained when the four regions are represented by vertices and the bridges by edges. This multigraph is shown in Figure 2.

The problem of traveling across every bridge without crossing any bridge more than once can be rephrased in terms of this model. The question becomes: Is there a simple circuit in this multigraph that contains every edge?

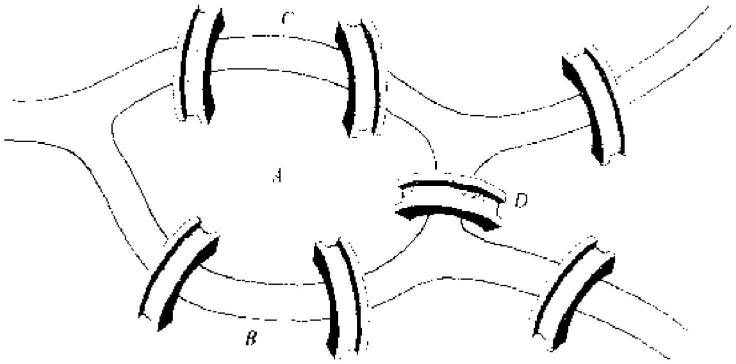


FIGURE 1 The Seven Bridges of Königsberg.

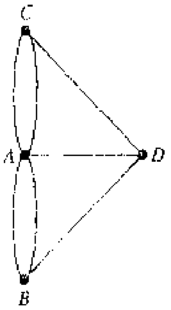


FIGURE 2 Multigraph Model of the Town of Königsberg.

DEFINITION 1. An *Euler circuit* in a graph G is a simple circuit containing every edge of G. An *Euler path* in G is a simple path containing every edge of G.

The following examples illustrate the concept of Euler circuits and paths.

EXAMPLE 1 Which of the undirected graphs in Figure 3 have an Euler circuit? Of those that do not, which have an Euler path?

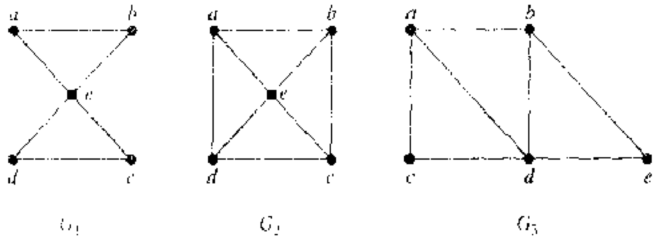


FIGURE 3 The Undirected Graphs G_1, G_2, and G_3.

Leonhard Euler (1707–1783). Leonhard Euler was the son of a Calvinist minister from the vicinity of Basel, Switzerland. At 13 he entered the University of Basel, pursuing a career in theology, as his father wished. At the university Euler was tutored by Johann Bernoulli of the famous Bernoulli family of mathematicians. His interest and skills led him to abandon his theological studies and take up mathematics. Euler obtained his master's degree in philosophy at the age of 16. In 1727 Peter the Great invited him to join the Academy at St. Petersburg. In 1741 he moved to the Berlin Academy, where he stayed until 1766. He then returned to St. Petersburg, where he remained for the rest of his life.

Euler was incredibly prolific, contributing to many areas of mathematics, including number theory, combinatorics, analysis, as well as its applications to such areas as music and naval architecture. He wrote over 1100 books and papers and left so much unpublished work that it took 47 years after he died for all his work to be published. During his life his papers accumulated so quickly that he kept a large pile of articles awaiting publication. The Berlin Academy published the papers on top of this pile so later results were often published before results they depended on or superseded. Euler had 13 children and was able to continue his work while a child or two bounced on his knees. He was blind for the last 17 years of his life, but because of his fantastic memory this did not diminish his mathematical output. The project of publishing his collected works, undertaken by the Swiss Society of Natural Science, is still going on and will require more than 75 volumes.

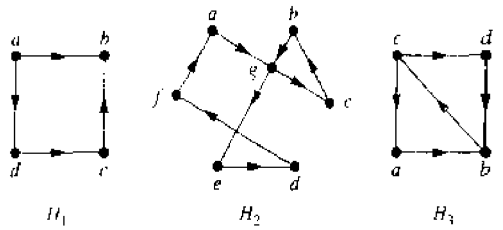


FIGURE 4 **The Directed Graphs H_1, H_2, and H_3.**

Solution: The graph G_1 has an Euler circuit, for example a, e, c, d, e, b, a. Neither of the graphs G_2 or G_3 has an Euler circuit (the reader should verify this). However, G_3 has an Euler path, namely, a, c, d, e, b, d, a, b. G_2 does not have an Euler path (as the reader should verify). ■

EXAMPLE 2 Which of the directed graphs in Figure 4 have an Euler circuit? Of those that do not, which have an Euler path?

Solution: The graph H_2 has an Euler circuit, for example, $a, g, c, b, g, e, d, f, a$. Neither H_1 nor H_3 has an Euler circuit (as the reader should verify). H_3 has an Euler path, namely c, a, b, c, d, b, but H_1 does not (as the reader should verify). ■

NECESSARY AND SUFFICIENT CONDITIONS FOR EULER CIRCUITS AND PATHS

There are simple criteria for determining whether a multigraph has an Euler circuit or an Euler path. Euler discovered them when he solved the famous Königsberg bridge problem. We will assume that all graphs discussed in this section have a finite number of vertices and edges.

What can we say if a connected multigraph has an Euler circuit? What we can show is that every vertex must have even degree. To do this, first note that an Euler circuit begins with a vertex a and continues with an edge incident to a, say $\{a, b\}$. The edge $\{a, b\}$ contributes 1 to $\deg(a)$. Each time the circuit passes through a vertex it contributes 2 to the vertex's degree, since the circuit enters via an edge incident with this vertex and leaves via another such edge. Finally, the circuit terminates where it started, contributing 1 to $\deg(a)$. Therefore, $\deg(a)$ must be even, because the circuit contributes 1 when it begins, 1 when it ends, and 2 every time it passes through a (if it ever does). A vertex other than a has even degree because the circuit contributes 2 to its degree each time it passes through the vertex. We conclude that if a connected graph has an Euler circuit, then every vertex must have even degree.

Is this necessary condition for the existence of an Euler circuit also sufficient? That is, must an Euler circuit exist in a connected multigraph if all vertices have even degree? This question can be settled affirmatively with a construction.

Suppose that G is a connected multigraph and the degree of every vertex of G is even. We will form a simple circuit that begins at an arbitrary vertex a of G. Let $x_0 = a$. First, we arbitrarily choose an edge $\{x_0, x_1\}$ incident with a. We continue by building a simple path $\{x_0, x_1\}, \{x_1, x_2\}, \ldots, \{x_{n-1}, x_n\}$ that is as long as possible. For instance, in the graph G in Figure 5 we begin at a and choose in succession the edges $\{a, f\}, \{f, c\}, \{c, b\}$, and $\{b, a\}$.

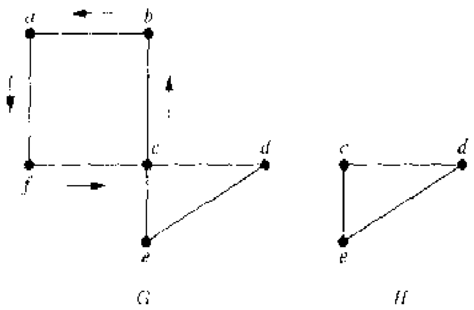


FIGURE 5 Constructing an Euler Circuit in G.

The path terminates since the graph has a finite number of edges. It begins at a with an edge of the form $\{a, x\}$, and it terminates at a with an edge of the form $\{y, a\}$. This follows because each time the path goes through a vertex with even degree, it uses only one edge to enter this vertex, so that at least one edge remains for the path to leave the vertex. This path may use all the edges, or it may not.

An Euler circuit has been constructed if all the edges have been used. Otherwise, consider the subgraph H obtained from G by deleting the edges already used and vertices that are not incident with any remaining edges. When we delete the circuit a, f, c, b, a from the graph in Figure 5, we obtain the subgraph labeled as H.

Since G is connected, H has at least one vertex in common with the circuit that has been deleted. Let w be such a vertex. (In our example, c is the vertex.)

Every vertex in H has even degree (because in G all vertices had even degree, and for each vertex, pairs of edges incident with this vertex have been deleted to form H). Note that H may not be connected. Beginning at w, construct a simple path in H by choosing edges as long as possible, as was done in G. This path must terminate at w. For instance, in Figure 5, c, d, e, c is a path in H. Next, form a circuit in G by splicing the circuit in H with the original circuit in G (this can be done since w is one of the vertices in this circuit). When this is done in the graph in Figure 5, we obtain the circuit a, f, c, d, e, c, b, a.

Continue this process until all edges have been used. (The process must terminate since there are only a finite number of edges in the graph.) This produces an Euler circuit. The construction shows that if the vertices of a connected multigraph all have even degree, then the graph has an Euler circuit.

We summarize these results in Theorem 1.

THEOREM 1 **A connected multigraph has an Euler circuit if and only if each of its vertices has even degree.**

We can now solve the Königsberg bridge problem. Since the multigraph representing these bridges, shown in Figure 2, has four vertices of odd degree, it does not have an Euler circuit. There is no way to start at a given point, cross each bridge exactly once, and return to the starting point.

Algorithm 1 gives the constructive procedure for finding Euler circuits given in the discussion preceding Theorem 1. (Since the circuits in the procedure are chosen arbitrarily, there is some ambiguity. We will not bother to remove this ambiguity by specifying the steps of the procedure more precisely.)

ALGORITHM 1 Constructing Euler Circuits.

```
procedure Euler(G: connected multigraph with all vertices of
      even degree)
circuit := a circuit in G beginning at an arbitrarily chosen
      vertex with edges successively added to form a path that
      returns to this vertex
H := G with the edges of this circuit removed
while H has edges
begin
      subcircuit := a circuit in H beginning at a vertex in H that
            also is an endpoint of an edge of circuit
      H := H with edges of subcircuit and all isolated vertices
            removed
      circuit := circuit with subcircuit inserted at the appropriate
            vertex
end {circuit is an Euler circuit}
```

The next example shows how Euler paths and circuits can be used to solve a type of puzzle.

EXAMPLE 3 Many puzzles ask you to draw a picture in a continuous motion without lifting a pencil so that no part of the picture is retraced. We can solve such puzzles using Euler circuits and paths. For example, can **Mohammed's scimitars,** shown in Figure 6, be drawn in this way, where the drawing begins and ends at the same point?

Solution: We can solve this problem since the graph G shown in Figure 6 has an Euler circuit. It has such a circuit since all its vertices have even degree. We will use Algorithm 1 to construct an Euler circuit. First, we form the circuit $a, b, d, c, b, e, i, f, e, a$. We obtain the subgraph H by deleting the edges in this circuit and all vertices that become isolated when these edges are removed. Then we form the circuit $d, g, h, j, i, h, k, g, f, d$ in H. After forming this circuit we have used all edges in G. Splicing this new circuit into the first circuit at the appropriate place produces the Euler circuit $a, b, d, g, h, j, i, h, k, g, f, d, c, b, e, i, f, e, a$. This circuit gives a way to draw the scimitars without lifting the pencil or retracing part of the picture. ■

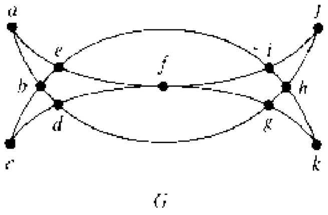


FIGURE 6 Mohammed's Scimitars.

Another algorithm for constructing Euler circuits, called Fleury's algorithm, is described in the exercises at the end of this section.

We will now show that a connected multigraph has an Euler path (and not an Euler circuit) if and only if it has exactly two vertices of odd degree. First, suppose that a connected multigraph does have an Euler path from a to b, but not an Euler circuit. The first edge of the path contributes 1 to the degree of a. A contribution of 2 to the degree of a is made every time the path passes through a. The last edge in the path contributes 1 to the degree of b. Every time the path goes through b there is a contribution of 2 to its degree. Consequently, both a and b have odd degree. Every other vertex has even degree, since the path contributes 2 to the degree of a vertex whenever it passes through it.

Now consider the converse. Suppose that a graph has exactly two vertices of odd degree, say a and b. Consider the larger graph made up of the original graph with the addition of an edge $\{a, b\}$. Every vertex of this larger graph has even degree, so that there is an Euler circuit. The removal of the new edge produces an Euler path in the original graph. The following theorem summarizes these results.

THEOREM 2 **A connected multigraph has an Euler path but not an Euler circuit if and only if it has exactly two vertices of odd degree.**

EXAMPLE 4 Which graphs shown in Figure 7 have an Euler path?

Solution: G_1 contains exactly two vertices of odd degree, namely, b and d. Hence, it has an Euler path that must have b and d as its endpoints. One such Euler path is d, a, b, c, d, b. Similarly, G_2 has exactly two vertices of odd degree, namely, b and d. So it has an Euler path that must have b and f as endpoints. One such Euler path is $b, a, g, f, e, d, c, g, b, c, f, d$. G_3 has no Euler path since it has six vertices of odd degree. ■

Returning to eighteenth-century Königsberg, is it possible to start at some point in the town, travel across all the bridges, and end up at some other point in town? This question can be answered by determining whether there is an Euler path in the multigraph representing the bridges in Königsberg. Since there are four vertices of odd degree in this multigraph, there is no Euler path, so such a trip is impossible.

Necessary and sufficient conditions for Euler paths and circuits in directed graphs are discussed in the exercises at the end of this section.

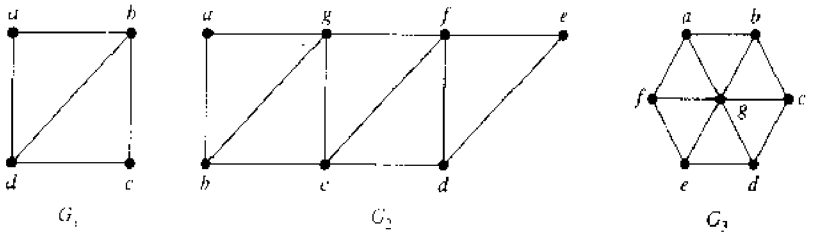


FIGURE 7 Three Undirected Graphs.

HAMILTON PATHS AND CIRCUITS

We have developed necessary and sufficient conditions for the existence of paths and circuits that contain every edge of a multigraph exactly once. Can we do the same for simple paths and circuits that contain every vertex of the graph exactly once?

DEFINITION 2. A path $x_0, x_1, \ldots, x_{n-1}, x_n$ in the graph $G = (V, E)$ is called a *Hamilton path* if $V = \{x_0, x_1, \ldots, x_{n-1}, x_n\}$ and $x_i \neq x_j$ for $0 \leq i < j \leq n$. A circuit $x_0, x_1, \ldots, x_{n-1}, x_n, x_0$ (with $n > 1$) in a graph $G = (V, E)$ is called a *Hamilton circuit* if $x_0, x_1, \ldots, x_{n-1}, x_n$ is a Hamilton path.

This terminology comes from a puzzle invented in 1857 by the Irish mathematician Sir William Rowan Hamilton. Hamilton's puzzle consisted of a wooden dodecahedron [a polyhedron with 12 regular pentagons as faces, as shown in Figure 8(a)], with a peg at each vertex of the dodecahedron, and string. The 20 vertices of the dodecahedron were labeled with different cities in the world. The object of the puzzle was to start at a city and travel along the edges of the dodecahedron, visiting each of the other 19 cities exactly once, and end back at the first city. The circuit traveled was marked off using the strings and pegs.

Since the author cannot supply each reader with a wooden solid with pegs and string, we will consider the equivalent question: Is there a circuit in the graph shown in Figure 8(b) that passes through each vertex exactly once? This solves the puzzle since this graph is isomorphic to the graph consisting of the vertices and edges of the dodecahedron. A solution of Hamilton's puzzle is shown in Figure 9.

---

web

William Rowan Hamilton (1805–1865). William Rowan Hamilton, the most famous Irish scientist ever to have lived, was born in 1805 in Dublin. His father was a successful lawyer, his mother came from a family noted for their intelligence, and he was a child prodigy. By the age of 3 he was an excellent reader and had mastered advanced arithmetic. Because of his brilliance, he was sent off to live with his uncle James, a noted linguist. By age 8 Hamilton had learned Latin, Greek, and Hebrew; by 10 he had also learned Italian and French and he began his study of oriental languages, including Arabic, Sanskrit, and Persian. During this period he took pride in knowing as many languages as his age. At 17, no longer devoted to learning new languages and having mastered calculus and much mathematical astronomy, he began original work in optics, and he also found an important mistake in Laplace's work on celestial mechanics. Before entering Trinity College, Dublin, at 18, Hamilton had not attended school; rather, he received private tutoring. At Trinity, he was a superior student in both the sciences and the classics. Prior to receiving his degree, because of his brilliance he was appointed the Astronomer Royal of Ireland, beating out several famous astronomers for the post. He held this position until his death, living and working at Dunsink Observatory outside of Dublin. Hamilton made important contributions to optics, abstract algebra, and dynamics. Hamilton invented algebraic objects called quaternions as an example of a noncommutative system. He discovered the appropriate way to multiply quaternions while walking along a canal in Dublin. In his excitement, he carved the formula in the stone of a bridge crossing the canal, a spot marked today by a plaque. Later, Hamilton remained obsessed with quaternions, working to apply them to other areas of mathematics, instead of moving to new areas of research.

In 1857 Hamilton invented "The Icosian Game" based on his work in noncommutative algebra. He sold the idea for 25 pounds to a dealer in games and puzzles. (Since the game never sold well, this turned out to be a bad investment for the dealer.) The "Traveler's Dodecahedron," also called "A Voyage Round the World," the puzzle described in this section, is a variant of that game.

Hamilton married his third love in 1833, but his marriage worked out poorly, since his wife, a semi-invalid, was unable to cope with his household affairs. He suffered from alcoholism and lived reclusively for the last two decades of his life. He died from gout in 1865, leaving masses of papers containing unpublished research. Mixed in with these papers were a large number of dinner plates, many containing the remains of desiccated, uneaten chops.

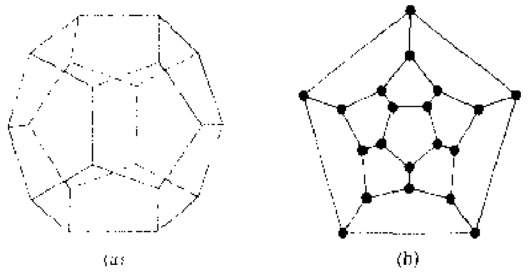


FIGURE 8 Hamilton's "Round the World" Puzzle.

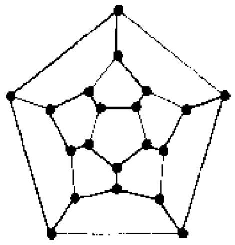


FIGURE 9 A Solution to the "Round the World" Puzzle.

EXAMPLE 5 Which of the simple graphs in Figure 10 have a Hamilton circuit or, if not, a Hamilton path?

Solution: G_1 has a Hamilton circuit: a, b, c, d, e, a. There is no Hamilton circuit in G_2 (this can be seen by noting that any circuit containing every vertex must contain the edge $\{a, b\}$ twice), but G_2 does have a Hamilton path, namely a, b, c, d. G_3 has neither a Hamilton circuit nor a Hamilton path, since any path containing all vertices must contain one of the edges $\{a, b\}$, $\{e, f\}$, and $\{c, d\}$ more than once. ■

Is there a simple way to determine whether a graph has a Hamilton circuit or path? At first, it might seem that there should be an easy way to determine this, since there is a simple way to answer the similar question of whether a graph has an Euler circuit. Surprisingly, there are no known simple necessary and sufficient criteria for the existence of Hamilton circuits. However, many theorems are known that give sufficient conditions for the existence of Hamilton circuits. Also, certain properties can be used to show that a graph has no Hamilton circuit. For instance, a graph with a vertex of degree 1 cannot have a Hamilton circuit, since in a Hamilton circuit each vertex is incident with two edges in the circuit. Moreover, if a vertex in the graph has degree 2, then both edges that are incident with this vertex must be part of any Hamilton circuit. Also, note that when a Hamilton circuit is being constructed and this circuit has passed through a vertex, then all remaining edges incident with this vertex, other than the two used in the circuit, can be removed from consideration. Furthermore, a Hamilton circuit cannot contain a smaller circuit within it.

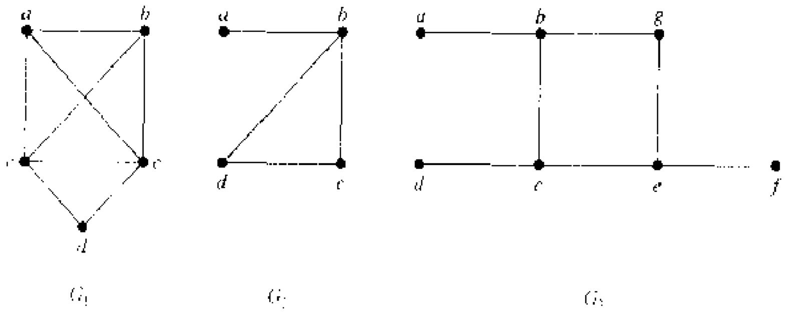


FIGURE 10 Three Simple Graphs.

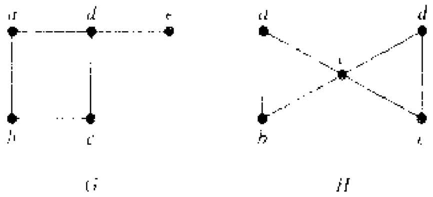


FIGURE 11 Two Graphs That Do Not Have a Hamilton Circuit.

EXAMPLE 6 Show that neither graph displayed in Figure 11 has a Hamilton circuit.

Solution: There is no Hamilton circuit in G since G has a vertex of degree 1, namely, e.

Now consider H. Since the degrees of the vertices a, b, d, and e are all 2, every edge incident with these vertices must be part of any Hamilton circuit. It is now easy to see that no Hamilton circuit can exist in H, for any Hamilton circuit would have to contain four edges incident with c, which is impossible. ■

EXAMPLE 7 Show that K_n has a Hamilton circuit whenever $n \geq 3$.

Solution: We can form a Hamilton circuit in K_n beginning at any vertex. Such a circuit can be built by visiting vertices in any order we choose, as long as the path begins and ends at the same vertex and visits each other vertex exactly once. This is possible since there are edges in K_n between any two vertices. ■

We now state a theorem that gives sufficient conditions for the existence of Hamilton circuits. This is just one of many such theorems known.

THEOREM 3 **If G is a connected simple graph with n vertices where $n \geq 3$, then G has a Hamilton circuit if the degree of each vertex is at least $n/2$.**

web

The best algorithms known for finding a Hamilton circuit in a graph or determining that no such circuit exists have exponential worst-case time complexity (in the number of vertices of the graph). Finding an algorithm that solves this problem with polynomial worst case time complexity would be a major accomplishment, since the existence of such an algorithm would imply that many other seemingly intractable problems could be solved using algorithms with polynomial worst-case time complexity.

We will now give an application of Hamilton circuits to coding.

EXAMPLE 8 Gray Codes The position of a rotating pointer can be represented in digital form. One way to do this is to split the circle into 2^n arcs of equal length and to assign a bit string of length n to each arc. Two ways to do this using bit strings of length three are shown in Figure 12.

The digital representation of the position of the pointer can be determined using a set of n contacts. Each contact is used to read one bit in the digital representation of the position. This is illustrated in Figure 13 for the two assignments from Figure 12.

When the pointer is near the boundary of two arcs, a mistake may be made in reading its position. This may result in a major error in the bit string read. For

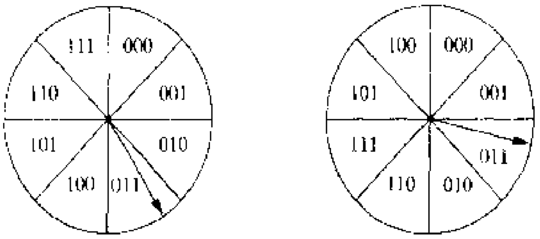

111
000
110
001
101
010
100
011
100
000
101
001
111
011
110
010

FIGURE 12 **Converting the Position of a Pointer into Digital Form.**

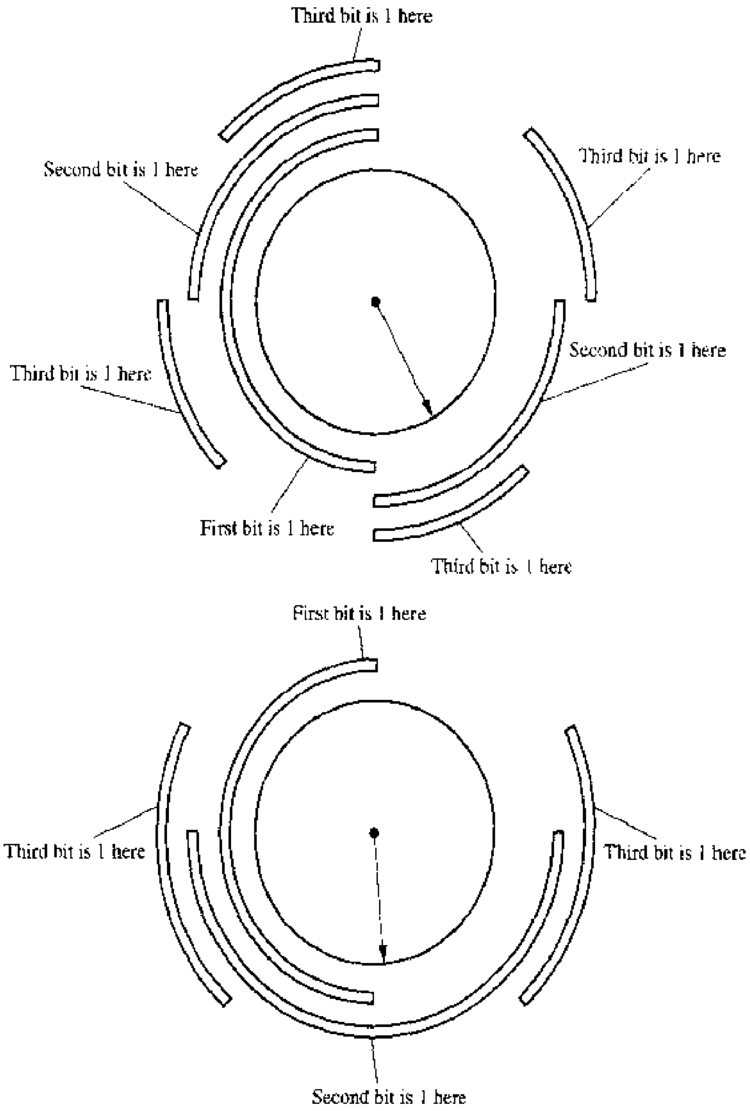

Third bit is 1 here
Second bit is 1 here
Third bit is 1 here
Second bit is 1 here
Third bit is 1 here
First bit is 1 here
Third bit is 1 here
First bit is 1 here
Third bit is 1 here
Third bit is 1 here
Second bit is 1 here

FIGURE 13 **The Digital Representation of the Position of the Pointer.**

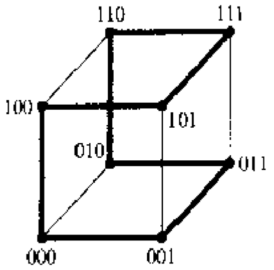


FIGURE 14 A Hamilton Circuit for Q_3.

instance, in the coding scheme in Figure 12(a), if a small error is made in determining the position of the pointer, the bit string 100 is read instead of 011. All three bits are incorrect! To minimize the effect of an error in determining the position of the pointer, the assignment of the bit strings to the 2^n arcs should be made so that only one bit is different in the bit strings represented by adjacent arcs. This is exactly the situation in the coding scheme in Figure 12(b). An error in determining the position of the pointer gives the bit string 010 instead of 011. Only one bit is wrong.

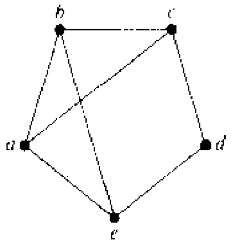
A **Gray code** is a labeling of the arcs of the circle so that adjacent arcs are labeled with bit strings that differ in exactly one bit. The assignment in Figure 12(b) is a Gray code. We can find a Gray code by listing all bit strings of length n in such a way that each string differs in exactly one position from the preceding bit string, and the last string differs from the first in exactly one position. We can model this problem using the n-cube Q_n. What is needed to solve this problem is a Hamilton circuit in Q_n. Such Hamilton circuits are easily found. For instance, a Hamilton circuit for Q_3 is displayed in Figure 14. The sequence of bit strings differing in exactly one bit produced by this Hamilton circuit is 000, 001, 011, 010, 110, 111, 101, 100.

Gray codes are named after Frank Gray, who invented them in the 1940s at AT&T Bell Laboratories to minimize the effect of errors in transmitting digital signals. ■

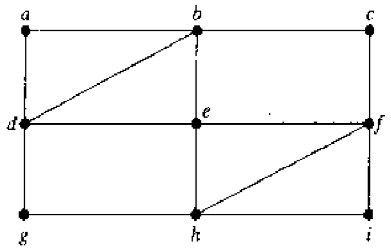
Exercises

In Exercises 1–7 determine whether each graph has an Euler circuit. Construct such a circuit when one exists.

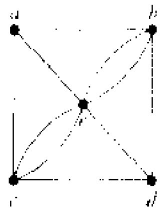
1.



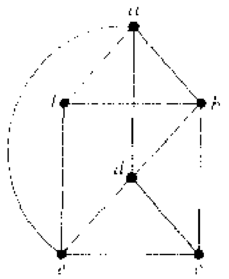
2.



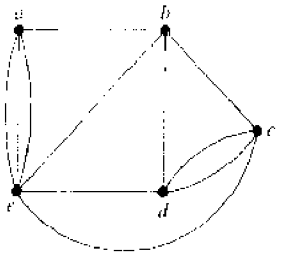
3.



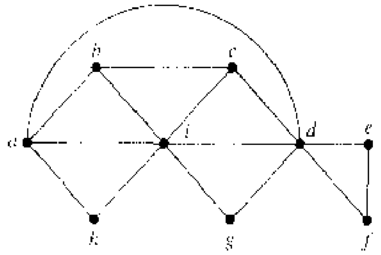
4.



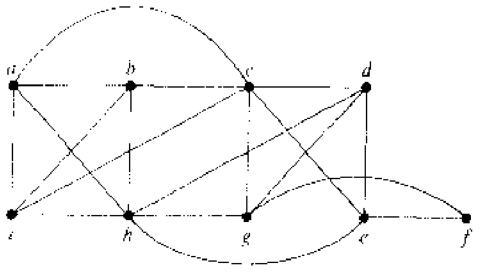
5.



6.



7.



8. Determine whether the graph in Exercise 1 has an Euler path. Construct such a path if it exists.

9. Determine whether the graph in Exercise 2 has an Euler path. Construct such a path if it exists.

10. Determine whether the graph in Exercise 3 has an Euler path. Construct such a path if it exists.

11. Determine whether the graph in Exercise 4 has an Euler path. Construct such a path if it exists.

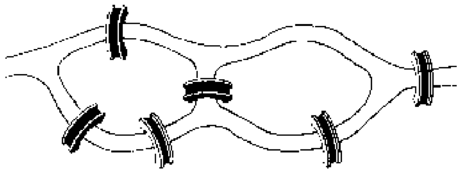
12. Determine whether the graph in Exercise 5 has an Euler path. Construct such a path if it exists.

13. Determine whether the graph in Exercise 6 has an Euler path. Construct such a path if it exists.

14. Determine whether the graph in Exercise 7 has an Euler path. Construct such a path if it exists.

15. In Kaliningrad (the Russian name for Königsberg) there are two additional bridges, besides the seven that were present in the 18th century. These new bridges connect regions *B* and *C* and regions *B* and *D*, respectively. Can someone cross all nine bridges in Kaliningrad exactly once and return to the starting point?

16. Can someone cross all the bridges shown in the following map exactly once and return to the starting point?

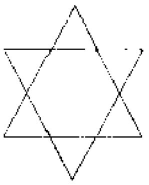


17. When can the center lines of the streets in a city be painted without traveling a street more than once? (Assume that all the streets are two-way streets.)

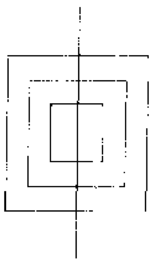
18. Devise a procedure, similar to Algorithm 1, for constructing Euler paths in multigraphs.

In Exercises 19–21 determine whether the picture shown can be drawn with a pencil in a continuous motion without lifting the pencil or retracing part of the picture.

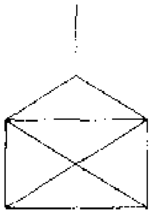
19.



20.



21.

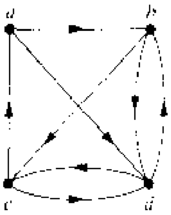


***22.** Show that a directed multigraph having no isolated vertices has an Euler circuit if and only if the graph is weakly connected and the in-degree and out-degree of each vertex are equal.

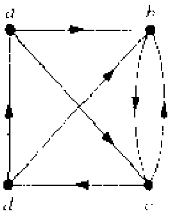
***23.** Show that a directed multigraph having no isolated vertices has an Euler path but not an Euler circuit if and only if the graph is weakly connected and the in-degree and out-degree of each vertex are equal for all but two vertices, one that has in-degree 1 larger than its out-degree and the other that has out-degree 1 larger than its in-degree.

In Exercises 24–28 determine whether the directed graph shown has an Euler circuit. Construct an Euler circuit if it exists.

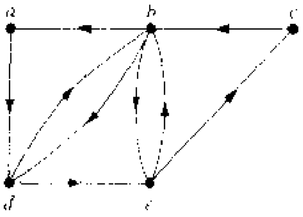
24.



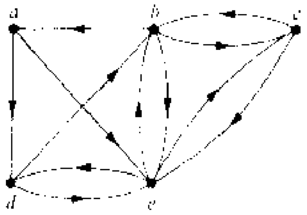
25.



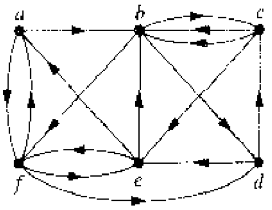
26.



27.



28.



29. Determine whether the directed graph in Exercise 24 has an Euler path. Construct an Euler path if one exists.

30. Determine whether the directed graph in Exercise 25 has an Euler path. Construct an Euler path if one exists.

31. Determine whether the directed graph in Exercise 26 has an Euler path. Construct an Euler path if one exists.

32. Determine whether the directed graph in Exercise 27 has an Euler path. Construct an Euler path if one exists.

33. Determine whether the directed graph in Exercise 28 has an Euler path. Construct an Euler path if one exists.

***34.** Devise an algorithm for constructing Euler circuits in directed graphs.

35. Devise an algorithm for constructing Euler paths in directed graphs.

36. For which values of n do the following graphs have an Euler circuit?

a) K_n **b)** C_n **c)** W_n **d)** Q_n

37. For which values of n do the graphs in Exercise 36 have an Euler path but no Euler circuit?

38. For which values of m and n does the complete bipartite graph $K_{m,n}$ have an

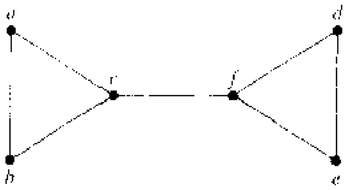
a) Euler circuit?

b) Euler path?

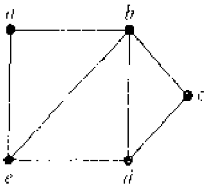
39. Find the least number of times it is necessary to lift a pencil from the paper when drawing each of the graphs in Exercises 1–7 without retracing any part of the graph.

In Exercises 40–46 determine whether the given graph has a Hamilton circuit. If it does, find such a circuit. If it does not, give an argument to show why no such circuit exists.

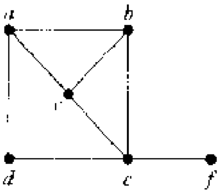
40.



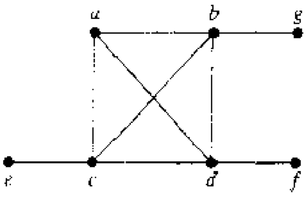
41.



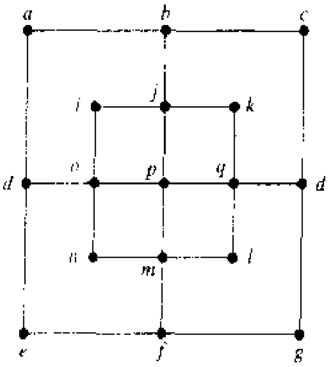
42.



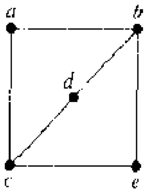
43.



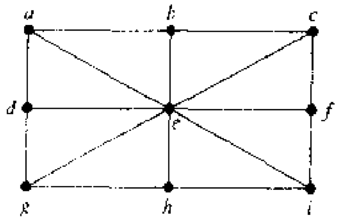
44.



45.



46.



47. Does the graph in Exercise 40 have a Hamilton path? If so, find such a path. If it does not, give an argument to show why no such path exists.

48. Does the graph in Exercise 41 have a Hamilton path? If so, find such a path. If it does not, give an argument to show why no such path exists.

49. Does the graph in Exercise 42 have a Hamilton path? If so, find such a path. If it does not, give an argument to show why no such path exists.

50. Does the graph in Exercise 43 have a Hamilton path? If so, find such a path. If it does not, give an argument to show why no such path exists.

*51. Does the graph in Exercise 44 have a Hamilton path? If so, find such a path. If it does not, give an argument to show why no such path exists.

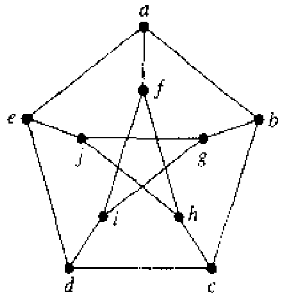
52. Does the graph in Exercise 45 have a Hamilton path? If so, find such a path. If it does not, give an argument to show why no such path exists.

*53. Does the graph in Exercise 46 have a Hamilton path? If so, find such a path. If it does not, give an argument to show why no such path exists.

54. For which values of n do the graphs in Exercise 36 have a Hamilton circuit?

55. For which values of m and n does the complete bipartite graph $K_{m,n}$ have a Hamilton circuit?

*56. Show that the Petersen graph, shown in the following diagram, does not have a Hamilton circuit, but that the subgraph obtained by deleting a vertex v, and all edges incident with v, does have a Hamilton circuit.



*57. Show that there is a Gray code of order n whenever n is a positive integer, or equivalently, show that the n-cube Q_n, $n > 1$, always has a Hamilton circuit. (*Hint:* Use mathematical induction. Show how to produce a Gray code of order n from one of order $n - 1$.)

Fleury's algorithm for constructing Euler circuits begins with an arbitrary vertex of a connected multigraph and forms a circuit by choosing edges successively. Once an edge is chosen, it is removed. Edges are chosen successively so that each edge begins where the last edge ends, and so that this edge is not a cut edge unless there is no alternative.

58. Use Fleury's algorithm to find an Euler circuit in the graph G in Example 5.

*59. Express Fleury's algorithm in pseudocode.

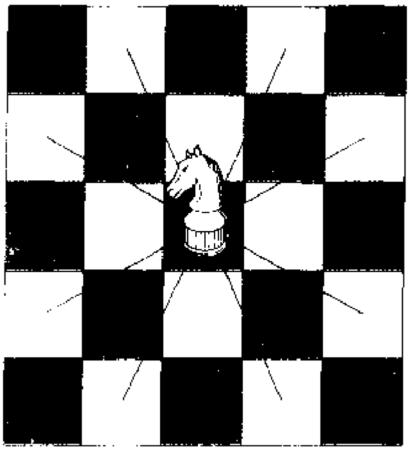
**60. Prove that Fleury's algorithm always produces an Euler circuit.

*61. Give a variant of Fleury's algorithm to produce Euler paths.

62. A diagnostic message can be sent out over a computer network to perform tests over all links and in all devices. What sort of paths should be used to test all links? To test all devices?

63. Show that a bipartite graph with an odd number of vertices does not have a Hamilton circuit.

A **knight** is a chess piece that can move either two spaces horizontally and one space vertically or one space horizontally and two spaces vertically. That is, a knight on square (x, y) can move to any of the eight squares $(x \pm 2, y \pm 1)$, $(x \pm 1, y \pm 2)$, if these squares are on the chessboard, as illustrated below.



A **knight's tour** is a sequence of legal moves by a knight starting at some square and visiting each square exactly once. A knight's tour is called **reentrant** if there is a legal move that takes the knight from the last square of the tour back to where the tour began. We can model knight's tours using the graph that has a vertex for each square on the board, with an edge connecting two vertices if a knight can legally move between the squares represented by these vertices.

64. Draw the graph that represents the legal moves of a knight on a 3×3 chessboard.

65. Draw the graph that represents the legal moves of a knight on a 3×4 chessboard.

66. **a)** Show that finding a knight's tour on an $m \times n$ chessboard is equivalent to finding a Hamilton path on the graph representing the legal moves of a knight on that board.

b) Show that finding a reentrant knight's tour on an $m \times n$ chessboard is equivalent to finding a Hamilton circuit on the corresponding graph.

***67.** Show that there is a knight's tour on a 3×4 chessboard.

***68.** Show that there is no knight's tour on a 3×3 chessboard.

***69.** Show that there is no knight's tour on a 4×4 chessboard.

70. Show that the graph representing the legal moves of a knight on an $m \times n$ chessboard, whenever m and n are positive integers, is bipartite.

71. Show that there is no reentrant knight's tour on an $m \times n$ chessboard when m and n are both odd. (*Hint:* Use Exercises 63, 66b, and 70.)

***72.** Show that there is a knight's tour on an 8×8 chessboard. (*Hint:* You can construct a knight's tour using the following method invented by Warnsdorff. Start in any square, and then always move to a square connected to the fewest number of unused squares. Although this method may not always produce a knight's tour, it often does.)

Julius Peter Christian Petersen (1839–1910). Julius Petersen was born in the Danish town of Sorø. His father was a dyer. In 1854 his parents were no longer able to pay for his schooling, so he became an apprentice in an uncle's grocery store. When this uncle died, he left Petersen enough money to return to school. After graduating, he began studying engineering at the Polytechnical School in Copenhagen, later deciding to concentrate on mathematics. He published his first textbook, a book on logarithms, in 1858. When his inheritance ran out, he had to teach to make a living. From 1859 until 1871 Petersen taught at a prestigious private high school in Copenhagen. While teaching high school he continued his studies, entering Copenhagen University in 1862. He married Laura Bertelsen in 1862; they had three children, two sons and a daughter.

Petersen obtained a mathematics degree from Copenhagen University in 1866 and finally obtained his doctorate in 1871 from that school. After receiving his doctorate, he taught at a polytechnic and military academy. In 1887 he was appointed to a professorship at the University of Copenhagen. Petersen was well-known in Denmark as the author of a large series of textbooks for high schools and universities. One of his books, *Methods and Theories for the Solution of Problems of Geometrical Construction*, was translated into eight languages, with the English language version last reprinted in 1960 and the French version reprinted as recently as 1990, more than a century after the original publication date.

Petersen worked in a wide range of areas, including algebra, analysis, cryptography, geometry, mechanics, mathematical economics, and number theory. His contributions to graph theory, including results on regular graphs, are his best-known work. He was noted for his clarity of exposition, problem-solving skills, originality, sense of humor, vigor, and teaching. One interesting fact about Petersen was that he preferred not to read the writings of other mathematicians. This led him often to rediscover results already proved by others, often with embarrassing consequences. However, he was often angry when other mathematicians did not read his writings!

Petersen's death was front page news in Copenhagen. A newspaper of the time described him as the Hans Christian Andersen of science—a child of the people who made good in the academic world.

7.6

Shortest Path Problems

INTRODUCTION

Many problems can be modeled using graphs with weights assigned to their edges. As an illustration, consider how an airline system can be modeled. We set up the basic graph model by representing cities by vertices and flights by edges. Problems involving distances can be modeled by assigning distances between cities to the edges.

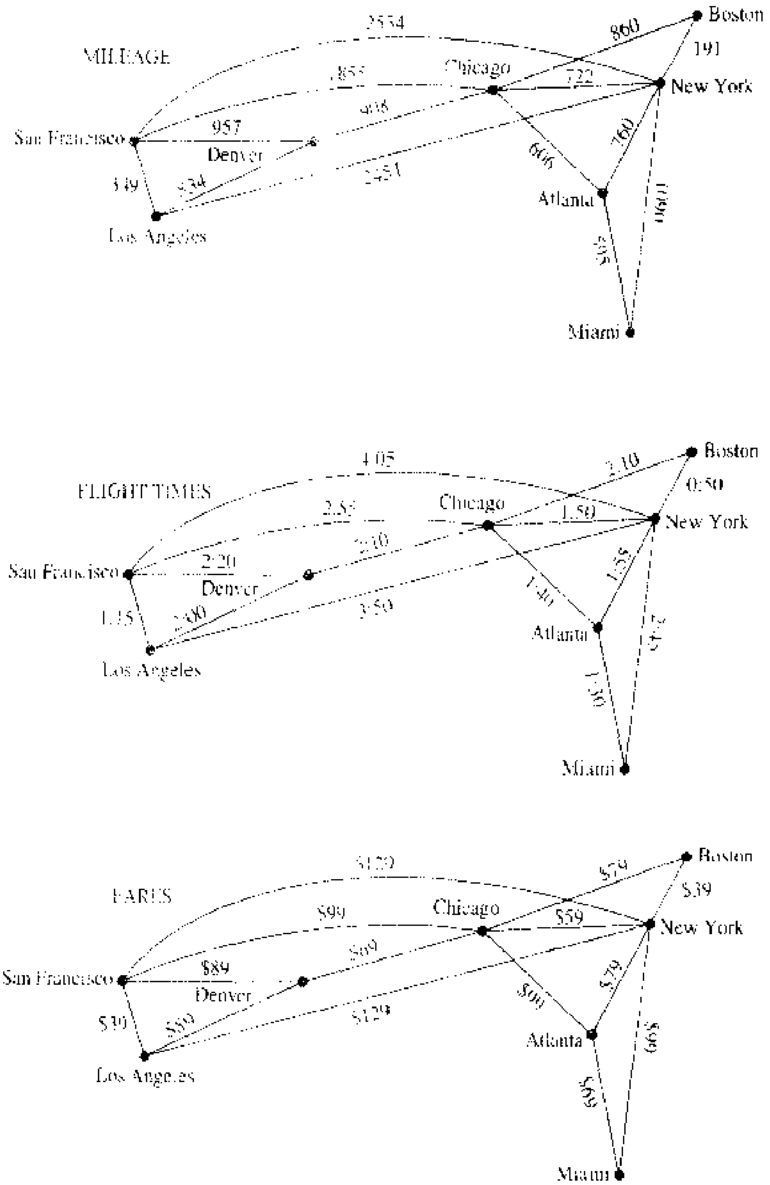
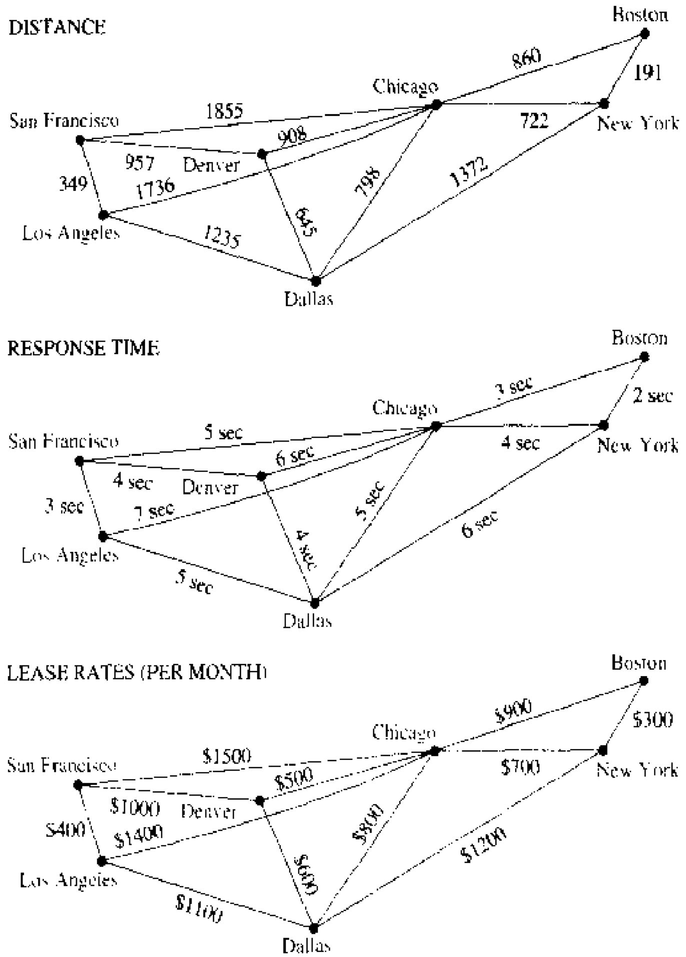

MILEAGE
2534
860
Boston
191
1855
Chicago
722
New York
908
957
San Francisco
Denver
760
606
2451
349
834
Atlanta
1090
Los Angeles
595
Miami
FLIGHT TIMES
4:05
2:10
Boston
0:50
2:55
Chicago
1:50
New York
2:10
San Francisco
2:20
Denver
1:55
1:40
1:15
2:00
3:50
Atlanta
2:45
Los Angeles
1:30
Miami
FARES
$129
$79
Boston
$39
$99
Chicago
$59
New York
$69
San Francisco
$89
Denver
$79
$69
$39
$69
$129
Atlanta
$99
Los Angeles
$69
Miami

FIGURE 1 Weighted Graphs Modeling an Airline System.



DISTANCE

Boston
860
191
Chicago
San Francisco
1855
722
New York
908
957
Denver
349
1736
798
1372
645
Los Angeles
1235
Dallas

RESPONSE TIME

Boston
3 sec
2 sec
Chicago
San Francisco
5 sec
4 sec
New York
6 sec
4 sec
Denver
3 sec
7 sec
5 sec
6 sec
4 sec
Los Angeles
5 sec
Dallas

LEASE RATES (PER MONTH)

Boston
$900
$300
Chicago
San Francisco
$1500
$700
New York
$500
$1000
Denver
$400
$1400
$800
$1200
$600
Los Angeles
$1100
Dallas

FIGURE 2 Weighted Graphs Modeling a Computer Network.

Problems involving flight time can be modeled by assigning flight times to edges. Problems involving fares can be modeled by assigning fares to the edges. Figure 1 displays three different assignments of weights to the edges of a graph representing distances, flight times, and fares, respectively.

Graphs that have a number assigned to each edge are called **weighted graphs.** Weighted graphs are used to model computer networks. Communications costs (such as the monthly cost of leasing a telephone line), the response times of the computers over these lines, or the distance between computers, can all be studied using weighted graphs. Figure 2 displays weighted graphs that represent three ways to assign weights to the edges of a graph of a computer network, corresponding to costs, response times over the lines, and distance.

Several types of problems involving weighted graphs arise frequently. Determining the path of least length between two vertices in a network is one such problem. To be more specific, let the **length** of a path in a weighted graph be the sum of the weights of the edges of this path. (The reader should note that this use of the term *length* is different from the use of *length* to denote the number of edges in a path in a graph without weights.) The question is: What is the shortest path, that is, the path of least length, between two given vertices? For instance, in the airline system represented by the weighted graph shown in Figure 1, what is the shortest path in air distance between Boston and Los Angeles? What combinations of flights has the smallest total flight time (that is, total time in the air, not including time between flights) between Boston and Los Angeles? What is the cheapest fare between these two cities? In the computer network shown in Figure 2, what is the least expensive set of telephone lines needed to connect the computers in San Francisco with those in New York? Which set of telephone lines gives the fastest response time for communications between San Francisco and New York? Which set of lines has the shortest overall distance?

Another important problem involving weighted graphs asks for the circuit of shortest total length that visits every vertex of a complete graph exactly once. This is the famous *traveling salesman problem* which asks for the order a salesman should visit each of the cities on his route exactly once so that he travels the minimum total distance. We will discuss the traveling salesman problem later in this section.

A SHORTEST PATH ALGORITHM

There are several different algorithms that find the shortest path between two vertices in a weighted graph. We will present an algorithm discovered by the Dutch mathematician E. Dijkstra in 1959. The version we will describe solves this problem in undirected weighted graphs where all the weights are positive. It is easy to adapt it to solve shortest path problems in directed graphs. ***web***

Before giving a formal presentation of the algorithm, we will give a motivating example.

EXAMPLE 1 What is the length of the shortest path between a and z in the weighted graph shown in Figure 3?

Solution: Although the shortest path is easily found by inspection, we will develop some ideas useful in understanding Dijkstra's algorithm. We will solve this problem by finding the length of the shortest path from a to successive vertices, until z is reached.

The only paths starting at a that contain no vertex other than a (until the terminal vertex is reached) are a, b and a, d. Since the lengths of a, b and a, d are 4 and 2, respectively, it follows that d is the closest vertex to a.

We can find the next closest vertex by looking at all paths that go through only a and d (until the terminal vertex is reached). The shortest such path to b is still a, b, with

---

Edsger Wybe Dijkstra (born 1930). Edsger Dijkstra, born in the Netherlands, began programming computers in the early 1950s while studying theoretical physics at the University of Leiden. In 1952, realizing that he was more interested in programming than in physics, he quickly completed the requirements for his physics degree and began his career as a programmer, even though programming was not a recognized profession. (In 1957, the authorities in Amsterdam refused to accept "programming" as his profession on his marriage license. However, they did accept "theoretical physicist" when he changed his entry to this.) ***web***

Dijkstra has been one of the most forceful proponents of programming as a scientific discipline. He has made fundamental contributions to the areas of operating systems, including deadlock avoidance; programming languages, including the notion of structured programming; and algorithms. In 1972 Dijkstra received the Turing Award from the Association for Computing Machinery, one of the most prestigious awards in computer science. Dijkstra became a Burroughs Research Fellow in 1973, and in 1984 he was appointed to a chair in Computer Science at the University of Texas, Austin.

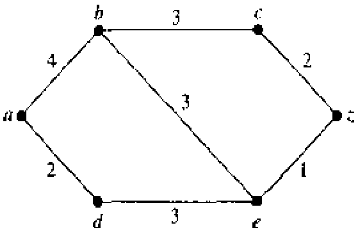


FIGURE 3 **A Weighted Simple Graph.**

length 4, and the shortest such path to e is a, d, e, with length 5. Consequently, the next closest vertex to a is b.

To find the third closest vertex to a, we need to examine only paths that go through only a, d, and b (until the terminal vertex is reached). There is a path of length 7 to c, namely, a, b, c, and a path of length 6 to z, namely, a, d, e, z. Consequently, z is the next closest vertex to a, and the length of the shortest path to z is 6. ■

Example 1 illustrates the general principles used in Dijkstra's algorithm. Note that the shortest path from a to z could have been found by inspection. However, inspection is impractical for both humans and computers for graphs with large numbers of edges.

We will now consider the general problem of finding the length of the shortest path between a and z in an undirected connected simple weighted graph. Dijkstra's algorithm proceeds by finding the length of the shortest path from a to a first vertex, the length of the shortest path from a to a second vertex, and so on, until the length of the shortest path from a to z is found.

The algorithm relies on a series of iterations. A distinguished set of vertices is constructed by adding one vertex at each iteration. A labeling procedure is carried out at each iteration. In this labeling procedure, a vertex w is labeled with the length of the shortest path from a to w that contains only vertices already in the distinguished set. The vertex added to the distinguished set is one with a minimal label among those vertices not already in the set.

We now give the details of Dijkstra's algorithm. It begins by labeling a with 0 and the other vertices with ∞. We use the notation $L_0(a) = 0$ and $L_0(v) = \infty$ for these labels before any iterations have taken place (the subscript 0 stands for the "0th" iteration). These labels are the lengths of the shortest paths from a to the vertices, where the paths contain only the vertex a. (Since no path from a to a vertex different from a exists, ∞ is the length of the shortest path between a and this vertex.)

Dijkstra's algorithm proceeds by forming a distinguished set of vertices. Let S_k denote this set after k iterations of the labeling procedure. We begin with $S_0 = \emptyset$. The set S_k is formed from S_{k-1} by adding a vertex u not in S_{k-1} with the smallest label. Once u is added to S_k, we update the labels of all vertices not in S_k, so that $L_k(v)$, the label of the vertex v at the kth stage, is the length of the shortest path from a to v that contains vertices only in S_k (that is, vertices that were already in the distinguished set together with u).

Let v be a vertex not in S_k. To update the label of v, note that $L_k(v)$ is the length of the shortest path from a to v containing only vertices in S_k. The updating can be carried out efficiently when the following observation is used: The shortest path from a

to v containing only elements of S_k is either the shortest path from a to v that contains only elements of S_{k-1} (that is, the distinguished vertices not including u), or it is the shortest path from a to u at the $(k-1)$st stage with the edge (u, v) added. In other words,

$$L_k(a, v) = \min\{L_{k-1}(a, v), L_{k-1}(a, u) + w(u, v)\}.$$

This procedure is iterated by successively adding vertices to the distinguished set until z is added. When z is added to the distinguished set, its label is the length of the shortest path from a to z. Dijkstra's algorithm is given in Algorithm 1. Later we will give a proof that this algorithm is correct.

ALGORITHM 1 **Dijkstra's Algorithm.**

```
procedure Dijkstra(G: weighted connected simple graph, with
        all weights positive)
{G has vertices $a = v_0, v_1, \ldots, v_n = z$ and weights $w(v_i, v_j)$
        where $w(v_i, v_j) = \infty$ if $\{v_i, v_j\}$ is not an edge in G}
for $i := 1$ to $n$
        $L(v_i) := \infty$
$L(a) := 0$
$S := \emptyset$
{the labels are now initialized so that the label of a is zero and all
        other labels are $\infty$, and S is the empty set}
while $z \notin S$
begin
        $u :=$ a vertex not in S with $L(u)$ minimal
        $S := S \cup \{u\}$
        for all vertices v not in S
                if $L(u) + w(u, v) < L(v)$ then $L(v) := L(u) + w(u, v)$
        {this adds a vertex to S with minimal label and updates the
                labels of vertices not in S}
end {$L(z)$ = length of shortest path from a to z}
```

The following example illustrates how Dijkstra's algorithm works. Afterward, we will show that this algorithm always produces the length of the shortest path between two vertices in a weighted graph.

EXAMPLE 2 Use Dijkstra's algorithm to find the length of the shortest path between the vertices a and z in the weighted graph displayed in Figure 4(a).

Solution: The steps used by Dijkstra's algorithm to find the shortest path between a and z are shown in Figure 4. At each iteration of the algorithm the vertices of the set S_k are circled. The shortest path from a to each vertex containing only vertices in S_k is indicated for each iteration. The algorithm terminates when z is circled. We find that the shortest path from a to z is a, c, b, d, e, z, with length 13. ■

Remark: In performing Dijkstra's algorithm it is sometimes more convenient to keep track of labels of vertices in each step using a table instead of redrawing the graph for each step.

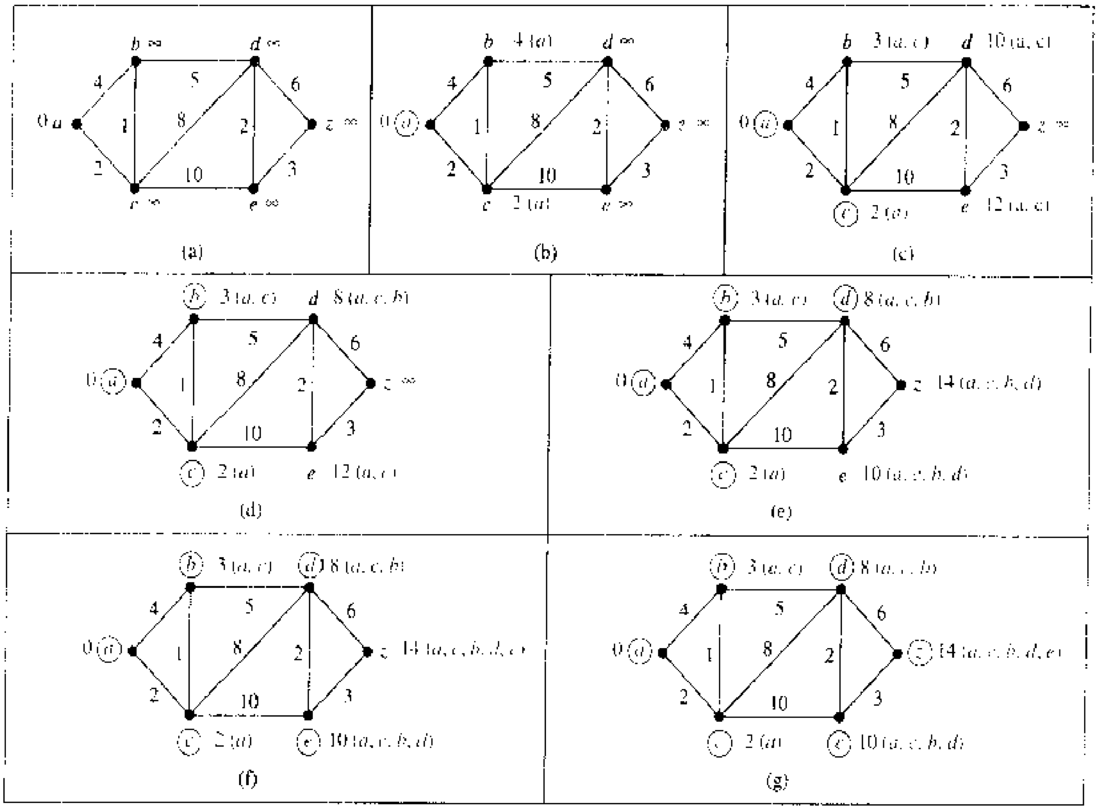


FIGURE 4 Using Dijkstra's Algorithm to Find the Shortest Path from a to z.

Next, we use an inductive argument to show that Dijkstra's algorithm produces the length of the shortest path between two vertices a and z in an undirected connected weighted graph. Take as the induction hypothesis the following assertion: At the kth iteration

(i) the label of a vertex v, $v \neq 0$, in S is the length of the shortest path from a to this vertex, and

(ii) the label of a vertex not in S is the length of the shortest path from a to this vertex that contains only (besides the vertex itself) vertices in S.

When $k = 0$, before any iterations are carried out, $S = \{a\}$, so the length of the shortest path from a to a vertex other than a is ∞, and the length of the shortest path from a to itself is 0 (here we are allowing a path to have no edges in it). Hence, the basis case is true.

Assume that the inductive hypothesis holds for the kth iteration. Let v be the vertex added to S at the $(k + 1)$st iteration so that v is a vertex not in S at the end of the kth iteration with the smallest label (in the case of ties, any vertex with smallest label may be used).

From the inductive hypothesis we see that the vertices in S before the $(k + 1)$st iteration are labeled with the length of the shortest path from a. Also, v must be labeled with the length of the shortest path to it from a. If this were not the case, at the end of the kth iteration there would be a path of length less than $L_k(v)$ containing a vertex not in S (because $L_k(v)$ is the length of the shortest path from a to v containing only vertices in S after the kth iteration). Let u be the first vertex not in S in such a path. There is a path with length less than $L_k(v)$ from a to u containing only vertices of S. This contradicts the choice of v. Hence, (*i*) holds at the end of the $(k+1)$st iteration.

Let u be a vertex not in S after $k+1$ iterations. A shortest path from a to u containing only elements of S either contains v or it does not. If it does not contain v, then by the inductive hypothesis its length is $L_k(u)$. If it does contain v, then it must be made up of a path from a to v of shortest possible length containing elements of S other than v, followed by the edge from v to u. In this case its length would be $L_k(v) + w(v, u)$. This shows that (*ii*) is true, since $L_{k+1}(u) = \min\{L_k(u), L_k(v) + w(v, u)\}$.

The following theorem has been proved.

THEOREM 1 **Dijkstra's algorithm finds the length of a shortest path between two vertices in a connected simple undirected weighted graph.**

We can now estimate the computational complexity of Dijkstra's algorithm (in terms of additions and comparisons). The algorithm uses no more than $n-1$ iterations, since one vertex is added to the distinguished set at each iteration. We are done if we can estimate the number of operations used for each iteration. We can identify the vertex not in S_k with the smallest label using no more than $n-1$ comparisons. Then we use an addition and a comparison to update the label of each vertex not in S_k. It follows that no more than $2(n-1)$ operations are used at each iteration, since there are no more than $n-1$ labels to update at each iteration. Since we use no more than $n-1$ iterations, each using no more than $2(n-1)$ operations, we have the following theorem.

THEOREM 2 **Dijkstra's algorithm uses $O(n^2)$ operations (additions and comparisons) to find the length of the shortest path between two vertices in a connected simple undirected weighted graph.**

THE TRAVELING SALESMAN PROBLEM

web

We now discuss an important problem involving weighted graphs. Consider the following problem: A traveling salesman wants to visit each of n cities exactly once and return to his starting point. For example, suppose that the salesman wants to visit Detroit, Toledo, Saginaw, Grand Rapids, and Kalamazoo (see Figure 5). In which order should he visit these cities to travel the minimum total distance? To solve this problem we can assume the salesman starts in Detroit (since this must be part of the circuit) and examine all possible ways for him to visit the other four cities and then return to Detroit (starting elsewhere will produce the same circuits). There are a total of 24 such circuits, but since we travel the same distance when we travel a circuit in reverse order, we need only consider 12 different circuits to find the minimum total distance he must travel. We list these 12 different circuits and the total distance traveled for each circuit. As can be seen from the list, the minimum total distance of 458 miles is traveled using the circuit Detroit–Toledo–Kalamazoo–Grand Rapids–Saginaw–Detroit (or its reverse).

We just described an instance of the **traveling salesman problem.** The traveling salesman problem asks for the circuit of minimum total weight in a weighted, complete,

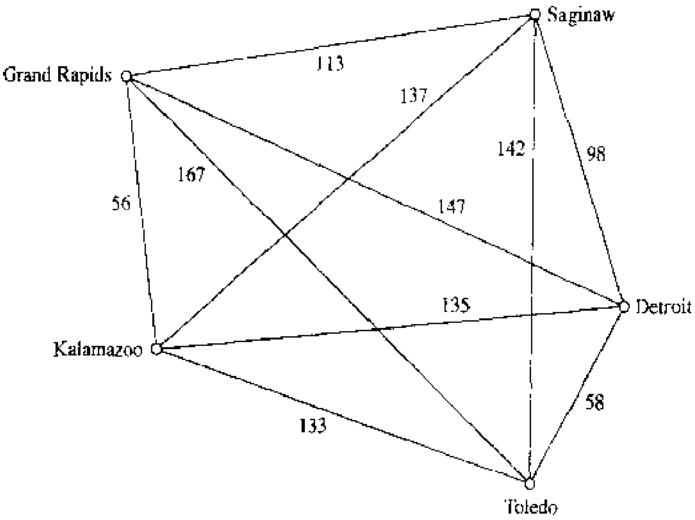


FIGURE 5 The Graph Showing the Distances Between Five Cities.

undirected graph that visits each vertex exactly once and returns to its starting point. This is equivalent to asking for a Hamilton circuit with minimum total weight in the complete graph, since each vertex is visited exactly once in the circuit.

The most straightforward way to solve an instance of the traveling salesman problem is to examine all possible Hamilton circuits and select one of minimum total length. How many circuits do we have to examine to solve the problem if there are n vertices in the graph? Once a starting point is chosen, there are $(n - 1!)$ different Hamilton circuits to examine, since there are $n - 1$ choices for the second vertex, $n - 2$ choices for the third vertex, and so on. Since a Hamilton circuit can be traveled in reverse order, we need only examine $(n - 1)!/2$ circuits to find our answer. Note that $(n - 1)!/2$ grows

Route	*Total Distance (miles)*
Detroit–Toledo–Grand Rapids–Saginaw–Kalamazoo–Detroit	610
Detroit–Toledo–Grand Rapids–Kalamazoo–Saginaw–Detroit	516
Detroit–Toledo–Kalamazoo–Saginaw–Grand Rapids–Detroit	588
Detroit–Toledo–Kalamazoo–Grand Rapids–Saginaw–Detroit	458
Detroit–Toledo–Saginaw–Kalamazoo–Grand Rapids–Detroit	540
Detroit–Toledo–Saginaw–Grand Rapids–Kalamazoo–Detroit	504
Detroit–Saginaw–Toledo–Grand Rapids–Kalamazoo–Detroit	598
Detroit–Saginaw–Toledo–Kalamazoo–Grand Rapids–Detroit	576
Detroit–Saginaw–Kalamazoo–Toledo–Grand Rapids–Detroit	682
Detroit–Saginaw–Grand Rapids–Toledo–Kalamazoo–Detroit	646
Detroit–Grand Rapids–Saginaw–Toledo–Kalamazoo–Detroit	670
Detroit–Grand Rapids–Toledo–Saginaw–Kalamazoo–Detroit	728

extremely rapidly. Trying to solve a traveling salesman problem in this way when there are only a few dozen vertices is impractical. For example, with 25 vertices, a total of $24!/2$ (approximately 3.1×10^{23}) different Hamilton circuits would have to be considered. If it took just 1 nanosecond (10^{-9} second) to examine each Hamilton circuit, a total of approximately 10 million years would be required to find a minimum-length Hamilton circuit in this graph.

Since the traveling salesman problem has both practical and theoretical importance, a great deal of effort has been devoted to devising efficient algorithms that solve it. However, no algorithm with polynomial worst-case time complexity is known for solving this problem. Furthermore, if a polynomial worst-case time complexity algorithm were discovered for the traveling salesman problem, many other difficult problems would also be solvable using polynomial worst-case time complexity algorithms (such as determining whether a proposition in n variables is a tautology, discussed in Chapter 1). This follows from the theory of NP-completeness. (For more information about this, consult the references on this topic at the end of the text.)

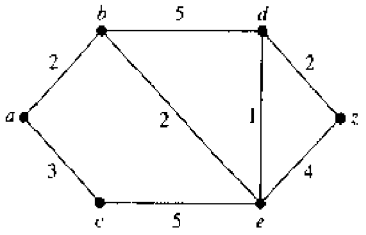
A practical approach to the traveling salesman problem when there are many vertices to visit is to use an **approximation algorithm.** These are algorithms that do not necessarily produce the exact solution to the problem but instead are guaranteed to produce a solution which is close to an exact solution. That is, they may produce a Hamilton circuit with total weight W' so that $W \leq W' \leq cW$, where W is the total length of an exact solution and c is a constant. For example, there is an algorithm with polynomial worst-case time complexity such that $c = 3/2$. In practice, algorithms have been developed that can solve traveling salesman problems with as many as 1000 vertices within 2% of an exact solution using only a few minutes of computer time. For more information about the traveling salesman problem, including history, applications, and algorithms, see the chapter on this topic in *Applications of Discrete Mathematics* [MiRo91].

Exercises

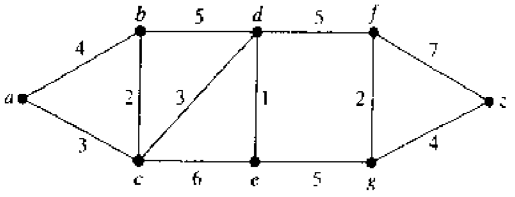
1. For each of the following problems about a subway system, describe a weighted graph model that can be used to solve the problem.
   a) What is the least amount of time required to travel between two stops?
   b) What is the minimum distance that can be traveled to reach a stop from another stop?
   c) What is the least fare required to travel between two stops if fares between stops are added to give the total fare?

In Exercises 2–4 find the length of the shortest path between a and z in the given weighted graph.

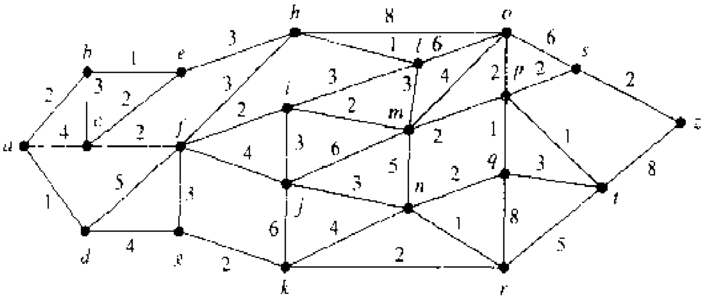
2.



3.



4.



5. What is the shortest path between a and z in each of the weighted graphs in Exercises 2–4?

6. Find the length of the shortest path between the following pairs of vertices in the weighted graph in Exercise 3.

a) a and d **b)** a and f **c)** c and f **d)** b and z

7. Find the shortest paths in the weighted graph in Exercise 3 between the pairs of vertices in Exercise 6.

8. Find the shortest path (in mileage) between each of the following pairs of cities in the airline system shown in Figure 1.

- **a)** New York and Los Angeles
- **b)** Boston and San Francisco
- **c)** Miami and Denver
- **d)** Miami and Los Angeles

9. Find the combination of flights with the least total air time between the pairs of cities in Exercise 8, using the flight times shown in Figure 1.

10. Find the least expensive combination of flights connecting the pairs of cities in Exercise 8, using the fares shown in Figure 1.

11. Find the shortest route (in distance) between computer centers in each of the following pairs of cities in the communications network shown in Figure 2.

- **a)** Boston and Los Angeles
- **b)** New York and San Francisco
- **c)** Dallas and San Francisco
- **d)** Denver and New York

12. Find the route with the shortest response time between the pairs of computer centers in Exercise 11 using the response times given in Figure 2.

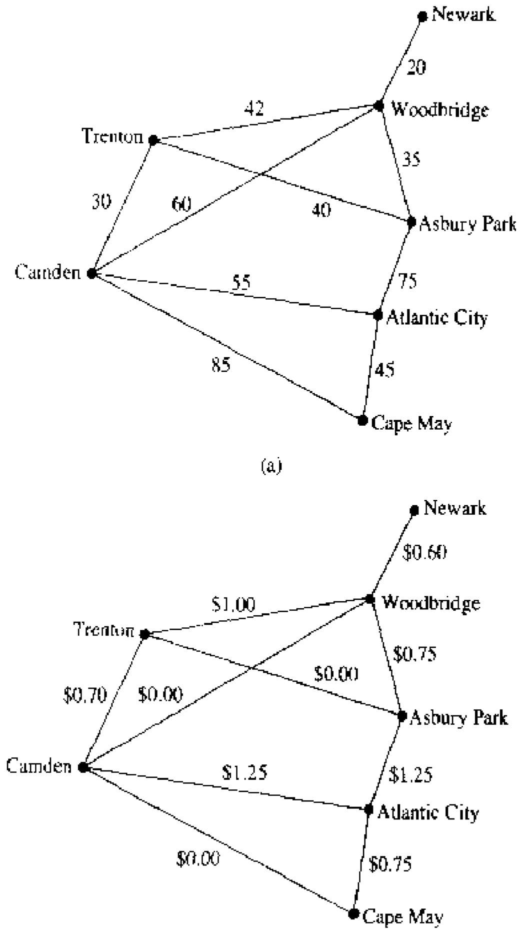
13. Find the least expensive route, in monthly lease charges, between the pairs of computer centers in Exercise 11 using the lease charges given in Figure 2.

14. Explain how to find the path with the least number of edges between two vertices in an undirected graph by considering it as a shortest path problem in a weighted graph.

15. Extend Dijkstra's algorithm for finding the length of the shortest path between two vertices in a weighted simple connected graph so that the length of the shortest path between the vertex a and every other vertex of the graph is found.

16. Extend Dijkstra's algorithm for finding the length of the shortest path between two vertices in a weighted simple connected graph so that the shortest path between these vertices is constructed.

17. The weighted graphs in the figures below show some major roads in New Jersey. Part (a) shows the distances between cities on these roads; part (b) shows the tolls.



(a)

(b)

a) Find the shortest route in distance between Newark and Camden, and between Newark and Cape May, using these roads.

ALGORITHM 2 Floyd's Algorithm.

```
procedure $Floyd(G$: weighted simple graph)
{$G$ has vertices $v_1, v_2, \ldots, v_n$ and weights $w(v_i, v_j)$ with $w(v_i, v_j) = \infty$ if $(v_i, v_j)$ is
not an edge}
for $i := 1$ to $n$
   for $j := 1$ to $n$
      $d(v_i, v_j) := w(v_i, v_j)$
for $i := 1$ to $n$
   for $j := 1$ to $n$
      for $k := 1$ to $n$
         if $d(v_j, v_i) + d(v_i, v_k) < d(v_j, v_k)$ then $d(v_j, v_k) :=$
            $d(v_j, v_i) + d(v_i, v_k)$
{$d(v_i, v_j)$ is the length of the shortest path between $v_i$ and $v_j$}
```

b) Find the least expensive route in terms of total tolls using the roads in the graph between the pairs of cities in part (a) of this exercise.

18. Is the shortest path between two vertices in a weighted graph unique if the weights of edges are distinct?

19. What are some applications where it is necessary to find the length of the longest simple path between two vertices in a weighted graph?

20. What is the length of the longest simple path in the weighted graph in Figure 4 between a and z? Between c and z?

Floyd's algorithm can be used to find the length of the shortest path between all pairs of vertices in a weighted connected simple graph. However, this algorithm cannot be used to construct shortest paths. (In the following, assign an infinite weight to any pair of vertices not connected by an edge in the graph.)

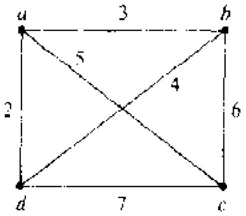
21. Use Floyd's algorithm to find the distance between all pairs of vertices in the weighted graph in Figure 4.

***22.** Prove that Floyd's algorithm determines the shortest distance between all pairs of vertices in a weighted simple graph.

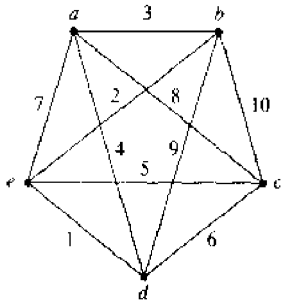
***23.** Give a big-O estimate of the number of operations (comparisons and additions) used by Floyd's algorithm to determine the shortest distance between every pair of vertices in a weighted simple graph with n vertices.

***24.** Show that Dijkstra's algorithm may not work if edges can have negative weights.

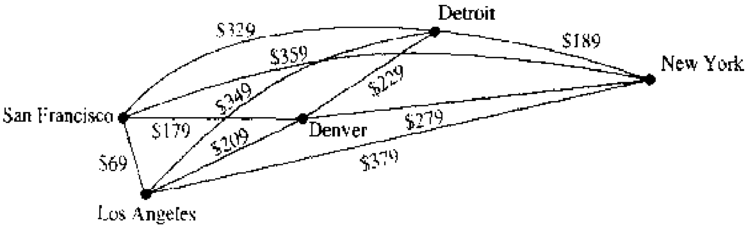
25. Solve the traveling salesman problem for the following graph by finding the total weight of all Hamilton circuits and determining a circuit with minimum total weight.



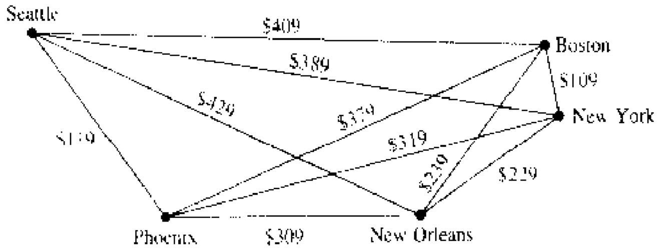
26. Solve the traveling salesman problem for the following graph by finding the total weight of all Hamilton circuits and determining a circuit with minimum total weight.



27. Find the route with the least total airfare that visits each of the cities in the following graph g where the weight on an edge is the least price available for a flight between the two cities.



28. Find the route with the least total airfare that visits each of the cities in the following graph g where the weights on an edge is the least price available for a flight between the two cities.



29. Construct a weighted undirected graph such that the total weight of a circuit which visits every vertex at least once is minimized for a circuit which visits some vertices more than once. (*Hint:* There are examples with three vertices.)

30. Show that the problem of finding the circuit of minimum total weight which visits every vertex of a weighted graph at least once can be reduced to the problem of finding the circuit of minimum total weight which visits each vertex of a weighted graph exactly once. Do so by constructing a new weighted graph with the same vertices and edges as the original graph but whose weight of the edge connecting the vertices u and v is equal to the minimum total weight of a path from u to v in the original graph.

7.7

Planar Graphs

INTRODUCTION

Consider the problem of joining three houses to each of three separate utilities, as shown in Figure 1. Is it possible to join these houses and utilities so that none of the connections cross? This problem can be modeled using the complete bipartite graph $K_{3,3}$. The original question can be rephrased as: Can $K_{3,3}$ be drawn in the plane so that no two of its edges cross?

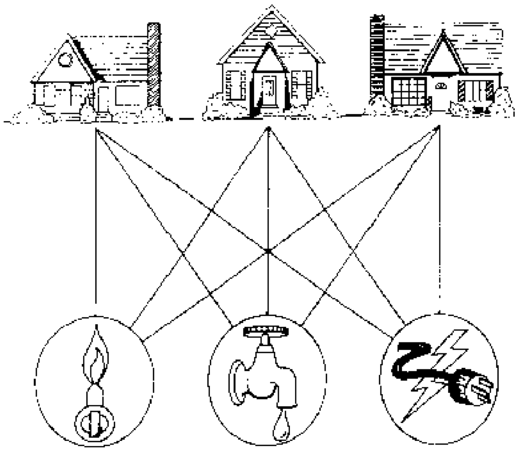


FIGURE 1 **Three Houses and Three Utilities.**

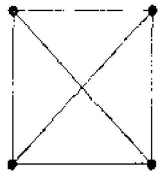


FIGURE 2 The Graph K_4.

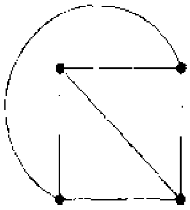


FIGURE 3 K_4 Drawn with No Crossings.

In this section we will study the question of whether a graph can be drawn in the plane without edges. In particular, we will answer the houses-and-utilities problem.

There are always many ways to represent a graph. When is it possible to find at least one way to represent this graph in a plane without any edges crossing?

DEFINITION 1. A graph is called *planar* if it can be drawn in the plane without any edges crossing (where a crossing of edges is the intersection of the lines or arcs representing them at a point other than their common endpoint). Such a drawing is called a *planar representation* of the graph.

A graph may be planar even if it is usually drawn with crossings, since it may be possible to draw it in a different way without crossings.

EXAMPLE 1 Is K_4 (shown in Figure 2 with two edges crossing) planar?

Solution: K_4 is planar because it can be drawn without crossings, as shown in Figure 3. ■

EXAMPLE 2 Is Q_3, shown in Figure 4, planar?

Solution: Q_3 is planar, because it can be drawn without any edges crossing, as shown in Figure 5. ■

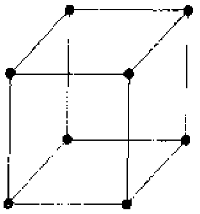


FIGURE 4 The Graph Q_3.

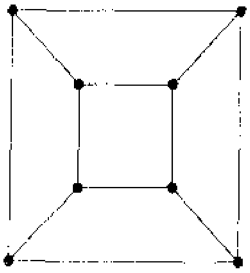


FIGURE 5 A Planar Representation.

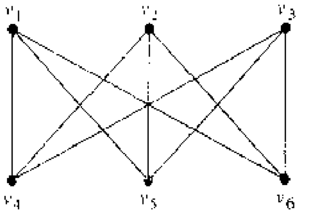


FIGURE 6 The Graph $K_{3,3}$.

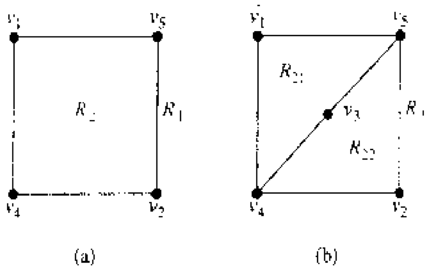


FIGURE 7 Showing $K_{3,3}$ Is Nonplanar.

We can show that a graph is planar by displaying a planar representation. It is harder to show that a graph is nonplanar. We will give an example to show how this can be done in an ad hoc fashion. Later we will develop some general results that can be used to do this.

EXAMPLE 3 Is $K_{3,3}$, shown in Figure 6, planar?

Solution: Any attempt to draw $K_{3,3}$ in the plane with no edges crossing is doomed. We now show why. In any planar representation of $K_{3,3}$, the vertices v_1 and v_2 must be connected to both v_4 and v_5. These four edges form a closed curve that splits the plane into two regions, R_1 and R_2, as shown in Figure 7(a). The vertex v_3 is in either R_1 or R_2. When v_3 is in R_2, the inside of the closed curve, the edges between v_3 and v_4 and between v_3 and v_5 separate R_2 into two subregions, R_{21} and R_{22}, as shown in Figure 7(b).

Next, note that there is no way to place the final vertex v_6 without forcing a crossing. For if v_6 is in R_1, then the edge between v_6 and v_3 cannot be drawn without a crossing. If v_6 is in R_{21}, then the edge between v_2 and v_6 cannot be drawn without a crossing. If v_6 is in R_{22}, then the edge between v_1 and v_6 cannot be drawn without a crossing.

A similar argument can be used when v_3 is in R_1. The completion of this argument is left for the reader (see Exercise 8 at the end of this section). It follows that $K_{3,3}$ is not planar. ■

Example 3 solves the utilities-and-houses problem that was described at the beginning of this section. The three houses and three utilities cannot be connected in the plane without a crossing. A similar argument can be used to show that K_5 is nonplanar. (See Exercise 9 at the end of this section.)

EULER'S FORMULA

A planar representation of a graph splits the plane into **regions,** including an unbounded region. For instance, the planar representation of the graph shown in Figure 8 splits the plane into six regions. These are labeled in the figure. Euler showed that all planar representations of a graph split the plane into the same number of regions. He accomplished this by finding a relationship among the number of regions, the number of vertices, and the number of edges of a planar graph.

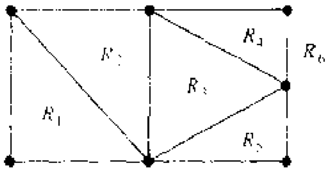


FIGURE 8 The Regions of the Planar Representation of a Graph.

THEOREM 1 EULER'S FORMULA **Let G be a connected planar simple graph with e edges and v vertices. Let r be the number of regions in a planar representation of G. Then $r = e - v + 2$.**

Proof: First, we specify a planar representation of G. We will prove the theorem by constructing a sequence of subgraphs $G_1, G_2, \ldots, G_e = G$, successively adding an edge at each stage. This is done using the following inductive definition. Arbitrarily pick one edge of G to obtain G_1. Obtain G_n from G_{n-1} by arbitrarily adding an edge that is incident with a vertex already in G_{n-1}, adding the other vertex incident with this edge if it is not already in G_{n-1}. This construction is possible since G is connected. G is obtained after e edges are added. Let r_n, e_n, and v_n represent the number of regions, edges, and vertices of the planar representation of G_n induced by the planar representation of G, respectively.

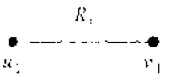
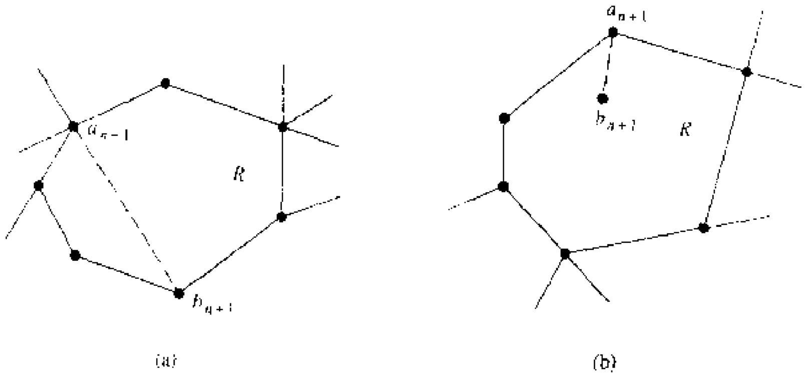


FIGURE 9 The Basis Case of the Proof of Euler's Formula.

The proof will now proceed by induction. The relationship $r_1 = e_1 - v_1 + 2$ is true for G_1, since $e_1 = 1$, $v_1 = 2$, and $r_1 = 1$. This is shown in Figure 9.

Now assume that $r_n = e_n - v_n + 2$. Let $\{a_{n+1}, b_{n+1}\}$ be the edge that is added to G_n to obtain G_{n+1}. There are two possibilities to consider. In the first case, both a_{n+1} and b_{n+1} are already in G_n. These two vertices must be on the boundary of a common region R, or else it would be impossible to add the edge $\{a_{n+1}, b_{n+1}\}$ to G_n without two edges crossing (and G_{n+1} is planar). The addition of this new edge splits R into two regions. Consequently, in this case, $r_{n+1} = r_n + 1$, $e_{n+1} = e_n + 1$, and $v_{n+1} = v_n$. Thus, each side of the formula relating the number of regions, edges, and vertices increases by exactly one, so this formula is still true. In other words, $r_{n+1} = e_{n+1} - v_{n+1} + 2$. This case is illustrated in Figure 10(a).

In the second case, one of the two vertices of the new edge is not already in G_n. Suppose that a_{n+1} is in G_n but that b_{n+1} is not. Adding this new edge does not produce



(a) (b)

FIGURE 10 Adding an Edge to G_n to Produce G_{n+1}.

any new regions, since b_{n+1} must be in a region that has a_{n+1} on its boundary. Consequently, $r_{n+1} = r_n$. Moreover, $e_{n+1} = e_n + 1$ and $v_{n+1} = v_n + 1$. Each side of the formula relating the number of regions, edges, and vertices remains the same, so the formula is still true. In other words, $r_{n+1} = e_{n+1} - v_{n+1} + 2$. This case is illustrated in Figure 10(b).

We have completed the induction argument. Hence $r_n = e_n - v_n + 2$ for all n. Since the original graph is the graph G_e, obtained after e edges have been added, the theorem is true. ☐

Euler's formula is illustrated in the following example.

EXAMPLE 4 Suppose that a connected planar simple graph has 20 vertices, each of degree 3. Into how many regions does a representation of this planar graph split the plane?

Solution: This graph has 20 vertices, each of degree 3, so that $v = 20$. Since the sum of the degrees of the vertices, $3v = 3 \cdot 20 = 60$, is equal to twice the number of edges, $2e$, we have $2e = 60$, or $e = 30$. Consequently, from Euler's formula, the number of regions is

$$r = e - v + 2 = 30 - 20 + 2 = 12. \qquad \blacksquare$$

Euler's formula can be used to establish some inequalities that must be satisfied by planar graphs. One such inequality is given in the following corollary.

COROLLARY 1 *If G is a connected planar simple graph with e edges and v vertices where $v \geq 3$, then $e \leq 3v - 6$.*

The proof of Corollary 1 is based on the concept of the **degree** of a region, which is defined to be the number of edges on the boundary of this region. When an edge occurs twice on the boundary (so that it is traced out twice when the boundary is traced out), it contributes 2 to the degree. The degrees of the regions of the graph shown in Figure 11 are displayed in the figure.

The proof of Corollary 1 can now be given.

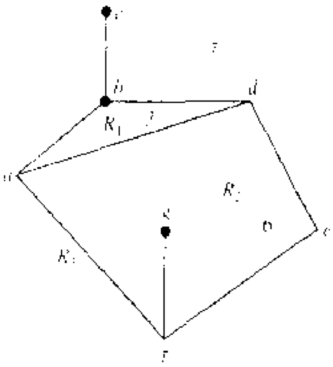


FIGURE 11 The Degrees of Regions.

Proof: A connected planar simple graph drawn in the plane divides the plane into regions, say r of them. The degree of each region is at least 3. (Since the graphs discussed here are simple graphs, no multiple edges that could produce regions of degree 2, or loops that could produce regions of degree 1, are permitted.) In particular, note that the degree of the unbounded region is at least 3 since there are at least three vertices in the graph.

Note that the sum of the degrees of the regions is exactly twice the number of edges in the graph, because each edge occurs on the boundary of a region exactly twice (either in two different regions, or twice in the same region). Since each region has degree greater than or equal to 3, it follows that

$$2e = \sum_{\text{all regions } R} \deg(R) \geq 3r.$$

Hence,

$$(2/3)e \geq r.$$

Using $r = e - v + 2$ (Euler's formula), we obtain

$$e - v + 2 \leq (2/3)e.$$

It follows that $e/3 \leq v - 2$. This shows that $e \leq 3v - 6$. □

This corollary can be used to demonstrate that K_5 is nonplanar.

EXAMPLE 5 Show that K_5 is nonplanar using Corollary 1.

Solution: The graph K_5 has five vertices and 10 edges. However, the inequality $e \leq 3v - 6$ is not satisfied for this graph since $e = 10$ and $3v - 6 = 9$. Therefore, K_5 is not planar. ■

It was previously shown that $K_{3,3}$ is not planar. Note, however, that this graph has six vertices and nine edges. This means that the inequality $e = 9 \leq 12 = 3 \cdot 6 - 6$ is satisfied. Consequently, the fact that the inequality $e \leq 3v - 6$ is satisfied does *not* imply that a graph is planar. However, the following corollary of Theorem 1 can be used to show that $K_{3,3}$ is nonplanar.

COROLLARY 2 If a connected planar simple graph has e edges and v vertices with $v \geq 3$ and no circuits of length 3, then $e \leq 2v - 4$.

The proof of Corollary 2 is similar to that of Corollary 1, except that in this case the fact that there are no circuits of length 3 implies that the degree of a region must be at least 4. The details of this proof are left for the reader (see Exercise 13 at the end of this section).

EXAMPLE 6 Use Corollary 2 to show that $K_{3,3}$ is nonplanar.

Solution: Since $K_{3,3}$ has no circuits of length 3 (this is easy to see since it is bipartite), Corollary 2 can be used. $K_{3,3}$ has six vertices and nine edges. Since $e = 9$ and $2v - 4 = 8$, Corollary 2 shows that $K_{3,3}$ is nonplanar. ■

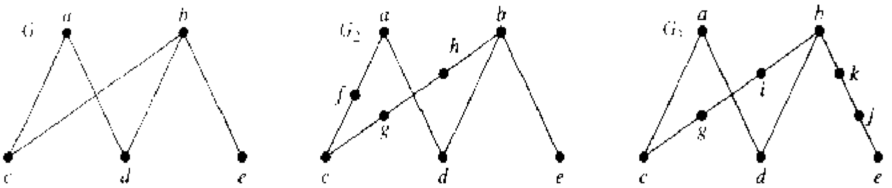


FIGURE 12 Homeomorphic Graphs.

KURATOWSKI'S THEOREM

We have seen that $K_{3,3}$ and K_5 are not planar. Clearly, a graph is not planar if it contains either of these two graphs as a subgraph. Furthermore, all nonplanar graphs must contain a subgraph that can be obtained from $K_{3,3}$ or K_5 using certain permitted operations.

If a graph is planar, so will be any graph obtained by removing an edge $\{u, v\}$ and adding a new vertex w together with edges $\{u, w\}$ and $\{w, v\}$. Such an operation is called an **elementary subdivision.** The graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are called **homeomorphic** if they can be obtained from the same graph by a sequence of elementary subdivisions. The three graphs displayed in Figure 12 are homeomorphic, since all can be obtained from the first graph by elementary subdivisions. (The reader should determine the sequences of elementary subdivisions needed to obtain G_2 and G_3 from G_1.)

The Polish mathematician Kuratowski established the following theorem in 1930, which characterizes planar graphs using the concept of graph homeomorphism.

THEOREM 2 A graph is nonplanar if and only if it contains a subgraph homeomorphic to $K_{3,3}$ or K_5.

It is clear that a graph containing a subgraph homeomorphic to $K_{3,3}$ or K_5 is nonplanar. However, the proof of the converse, namely that every nonplanar graph contains a subgraph homeomorphic to $K_{3,3}$ or K_5, is complicated and will not be given here. The following examples illustrate how Kuratowski's theorem is used.

EXAMPLE 7 Determine whether the graph G shown in Figure 13 is planar.

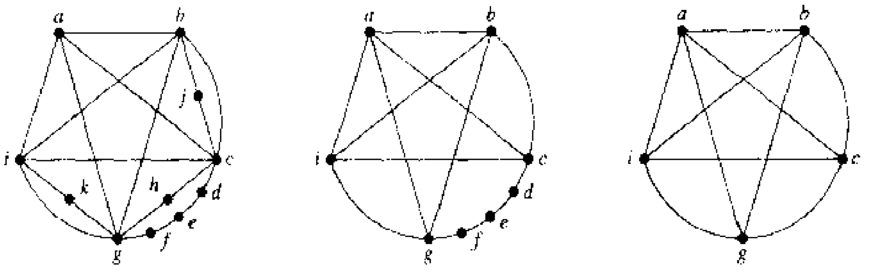
Solution: G has a subgraph H homeomorphic to K_5. H is obtained by deleting h, j, and k and all edges incident with these vertices. H is homeomorphic to K_5 since it can be

---

web

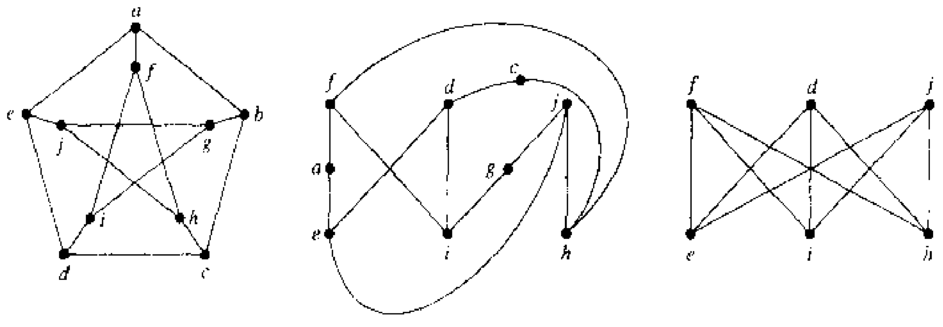
Kazimierz Kuratowski (1896–1980). Kazimierz Kuratowski, the son of a famous Warsaw lawyer, attended secondary school in Warsaw. He studied in Glasgow, Scotland, from 1913 to 1914 but could not return there after the outbreak of World War I. In 1915 he entered Warsaw University, where he was active in the Polish patriotic student movement. He published his first paper in 1919 and received his Ph.D. in 1921. He was an active member of the group known as the Warsaw School of Mathematics, working in the areas of the foundations of set theory and topology. He was appointed associate professor at the Lwów Polytechnical University, where he stayed for 7 years, collaborating with the important Polish mathematicians Banach and Ulam. In 1930, while at Lwów, Kuratowski completed his work characterizing planar graphs.

In 1934 he returned to Warsaw University as a full professor. Until the start of World War II, he was active in research and teaching. During the war, because of the persecution of educated Poles, Kuratowski went into hiding under an assumed name and taught at the clandestine Warsaw University. After the war he helped revive Polish mathematics, serving as director of the Polish National Mathematics Institute. He wrote over 180 papers and three widely used textbooks.



G H K_5

FIGURE 13 The Undirected Graph G, a Subgraph H Homeomorphic to K_5, and K_5.



(a) (b) H (c) $K_{3,3}$

FIGURE 14 (a) The Petersen Graph, (b) a Subgraph H Homeomorphic to $K_{3,3}$, and (c) $K_{3,3}$.

obtained from K_5 (with vertices a, b, c, g, and i) by a sequence of elementary subdivisions, adding the vertices d, e, and f. (The reader should construct such a sequence of elementary subdivisions.) Hence, G is nonplanar. ■

EXAMPLE 8 Is the Petersen graph, shown in Figure 14(a), planar? (The Danish mathematician Julius Petersen introduced this graph in 1891; it is often used to illustrate various theoretical properties of graphs.)

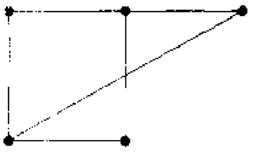
Solution: The subgraph H of the Petersen graph obtained by deleting b and the three edges that have b as an endpoint, shown in Figure 14(b), is homeomorphic to $K_{3,3}$, with vertex sets $\{f, d, j\}$ and $\{e, i, h\}$, since it can be obtained by a sequence of elementary subdivisions, deleting $\{d, h\}$ and adding $\{c, h\}$ and $\{c, d\}$, deleting $\{e, f\}$ and adding $\{a, e\}$ and $\{a, f\}$, and deleting $\{i, j\}$ and adding $\{g, i\}$ and $\{g, j\}$. Hence, the Petersen graph is not planar. ■

Exercises

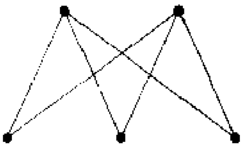
1. Can five houses be connected to two utilities without connections crossing?

In Exercises 2–4 draw the given planar graph without any crossings.

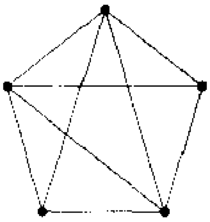
2.



3.

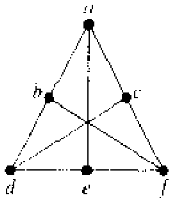


4.

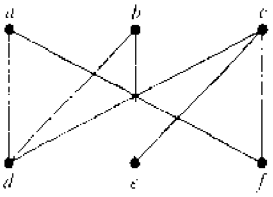


In Exercises 5–7 determine whether the given graph is planar. If so, draw it so that no edges cross.

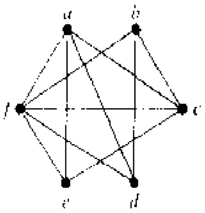
5.



6.



7.



8. Complete the argument in Example 3.

9. Show that K_5 is nonplanar using an argument similar to that given in Example 3.

10. Suppose that a connected planar graph has eight vertices, each of degree 3. Into how many regions is the plane divided by a planar representation of this graph?

11. Suppose that a connected planar graph has six vertices, each of degree 4. Into how many regions is the plane divided by a planar representation of this graph?

12. Suppose that a connected planar graph has 30 edges. If a planar representation of this graph divides the plane into 20 regions, how many vertices does this graph have?

13. Prove Corollary 2.

14. Suppose that a connected bipartite planar simple graph has e edges and v vertices. Show that $e \leq 2v - 4$ if $v \geq 3$.

***15.** Suppose that a connected planar simple graph with e edges and v vertices contains no simple circuits of length 4 or less. Show that $e \leq (5/3)v - (10/3)$ if $v \geq 4$.

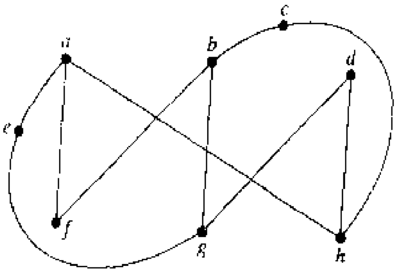
16. Suppose that a planar graph has k connected components, e edges, and v vertices. Also suppose that the plane is divided into r regions by a planar representation of the graph. Find a formula for r in terms of e, v, and k.

17. Which of the following nonplanar graphs have the property that the removal of any vertex and all edges incident with that vertex produces a planar graph?

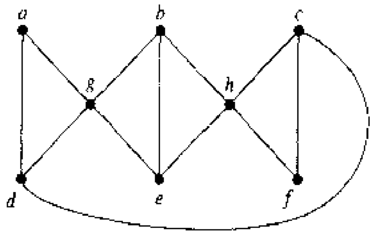
a) K_5 **b)** K_6 **c)** $K_{3,3}$ **d)** $K_{3,4}$

In Exercises 18–20 determine whether the given graph is homeomorphic to $K_{3,3}$.

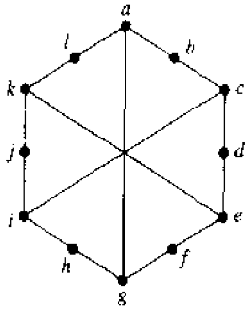
18.



19.

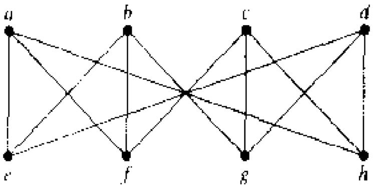


20.

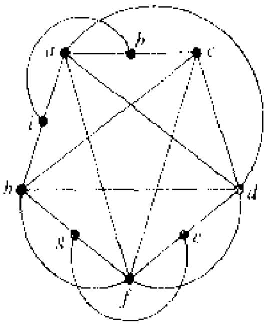


In Exercises 21–23 use Kuratowski's theorem to determine whether the given graph is planar.

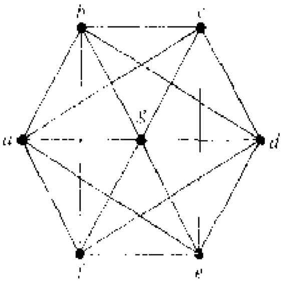
21.



22.



23.



eb The **crossing number** of a simple graph is the minimum number of crossings that can occur when this graph is drawn in the plane where no three arcs representing edges are permitted to cross at the same point.

24. Show that $K_{3,3}$ has 1 as its crossing number.

****25.** Find the crossing numbers of each of the following nonplanar graphs.

a) K_5 **b)** K_6 **c)** K_7
d) $K_{3,4}$ **e)** $K_{4,4}$ **f)** $K_{5,5}$

***26.** Find the crossing number of the Petersen graph.

****27.** Show that if m and n are even positive integers, the crossing number of $K_{m,n}$ is less than or equal to $mn(m-2)(n-2)/16$. (*Hint:* Place m vertices along the x-axis so that they are equally spaced and symmetric about the origin and place n vertices along the y-axis so that they are equally spaced and symmetric about the origin. Now connect each of the m vertices on the x-axis to each of the vertices on the y-axis and count the crossings.)

The **thickness** of a simple graph G is the smallest number of planar subgraphs of G that have G as their union.

28. Show that $K_{3,3}$ has 2 as its thickness.

***29.** Find the thickness of the graphs in Exercise 25.

30. Show that if G is a connected simple graph with v vertices and e edges, then the thickness of G is at least $\lceil e/(3v-6)\rceil$.

***31.** Use Exercise 30 to show that the thickness of K_n is at least $\lfloor (n+7)/6 \rfloor$ whenever n is a positive integer.

32. Show that if G is a connected simple graph with v vertices and e edges and no circuits of length three, then the thickness of G is at least $\lceil e/(2v-4)\rceil$.

33. Use Exercise 32 to show that the thickness of $K_{m,n}$ is at least $\lceil mn/(2m+2n-4)\rceil$ whenever m and n are positive integers.

***34.** Draw K_5 on the surface of a torus (a doughnut-shaped solid) so no edges cross.

***35.** Draw $K_{3,3}$ on the surface of a torus so no edges cross.

7.8

Graph Coloring

INTRODUCTION

Problems related to the coloring of maps of regions, such as maps of parts of the world, *web* have generated many results in graph theory. When a map* is colored, two regions with a common border are customarily assigned different colors. One way to ensure

---

*We will assume that all regions in a map are connected. This eliminates any problems presented by such geographical entities as Michigan.

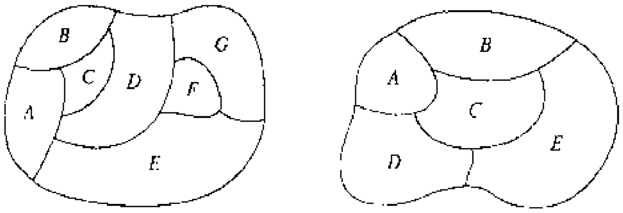


FIGURE 1 **Two Maps.**

that two adjacent regions never have the same color is to use a different color for each region. However, this is inefficient, and on maps with many regions it would be hard to distinguish similar colors. Instead, a small number of colors should be used whenever possible. Consider the problem of determining the least number of colors that can be used to color a map so that adjacent regions never have the same color. For instance, for the map shown on the left in Figure 1, four colors suffice, but three colors are not enough. (The reader should check this.) In the map on the right in Figure 1, three colors are sufficient (but two are not).

Each map in the plane can be represented by a graph. To set up this correspondence, each region of the map is represented by a vertex. Edges connect two vertices if the regions represented by these vertices have a common border. Two regions that touch at only one point are not considered adjacent. The resulting graph is called the **dual graph** of the map. By the way in which dual graphs of maps are constructed, it is clear that any map in the plane has a planar dual graph. Figure 2 displays the dual graphs that correspond to the maps shown in Figure 1.

The problem of coloring the regions of a map is equivalent to the problem of coloring the vertices of the dual graph so that no two adjacent vertices in this graph have the same color. We give the following definition.

DEFINITION 1. **A *coloring* of a simple graph is the assignment of a color to each vertex of the graph so that no two adjacent vertices are assigned the same color.**

A graph can be colored by assigning a different color to each of its vertices. However, for most graphs a coloring can be found that uses fewer colors than the number of vertices in the graph. What is the least number of colors necessary?

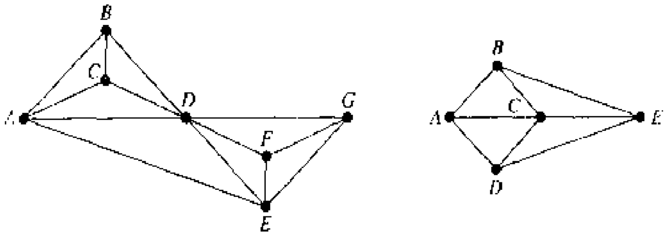


FIGURE 2 **Dual Graphs of the Maps in Figure 1.**

DEFINITION 2. **The *chromatic number* of a graph is the least number of colors needed for a coloring of this graph.**

Note that asking for the chromatic number of a planar graph is the same as asking for the minimum number of colors required to color a planar map so that no two adjacent regions are assigned the same color. This question has been studied for more than 100 years. The answer is provided by one of the most famous theorems in mathematics.

THEOREM 1 **The Four Color Theorem** **The chromatic number of a planar graph is no greater than four.**

The four color theorem was originally posed as a conjecture in the 1850s. It was finally proved by the American mathematicians Kenneth Appel and Wolfgang Haken in 1976. Prior to 1976, many incorrect proofs were published, often with hard-to-find errors. In addition, many futile attempts were made to construct counterexamples by drawing maps that require more than four colors.

Perhaps the most notorious fallacious proof in all of mathematics is an incorrect proof of the four color theorem published in 1879 by a London barrister and amateur mathematician, Alfred Kempe. Mathematicians accepted his proof as correct until 1890, when Percy Heawood found an error that made Kempe's argument incomplete. However, Kempe's line of reasoning turned out to be the basis of the successful proof given by Appel and Haken. Their proof relies on a careful case-by-case analysis carried out by computer. They showed that if the four color theorem were false, there would have to be a counterexample of one of approximately 2000 different types, and they then showed that none of these types could lead to a counterexample. They used over 1000 hours of computer time in their proof. This proof generated a large amount of controversy, since computers played such an important role in it. For example, could there be an error in a computer program that led to incorrect results? Was their argument really a proof if it depended on what could be unreliable computer output?

The four color theorem applies only to planar graphs. Nonplanar graphs can have arbitrarily large chromatic numbers, as will be shown in Example 2.

Two things are required to show that the chromatic number of a graph is n. First, we must show that the graph can be colored with n colors. This can be done by constructing

---

Alfred Bray Kempe (1849–1922). Kempe was a barrister and a leading authority on ecclesiastical law. However, having studied mathematics at Cambridge University, he retained his interest in it, and later in life he devoted considerable time to mathematical research. Kempe made contributions to kinematics, the branch of mathematics dealing with motion, and to mathematical logic. However, Kempe is best remembered for his fallacious proof of the four color theorem.

Historical Note: In 1852, an ex-student of Augustus De Morgan, Francis Guthrie, noticed that the counties in England could be colored using four colors so that no adjacent counties were assigned the same color. On this evidence, he conjectured that the four color theorem was true. Francis told his brother Frederick, at that time a student of De Morgan, about this problem. Frederick in turn asked his teacher De Morgan about his brother's conjecture. De Morgan was extremely interested in this problem and publicized it throughout the mathematical community. In fact, the first written reference to the conjecture can be found in a letter from De Morgan to Sir William Rowan Hamilton. Although De Morgan thought Hamilton would be interested in this problem, Hamilton apparently was not interested in it, since it had nothing to do with quaternions.

Historical Note: Although a simpler proof of the four color theorem was found by Robertson, Sanders, Seymour, and Thomas in 1996, reducing the computational part of the proof to examining 633 configurations, no proof that does not rely on extensive computation has yet been found.

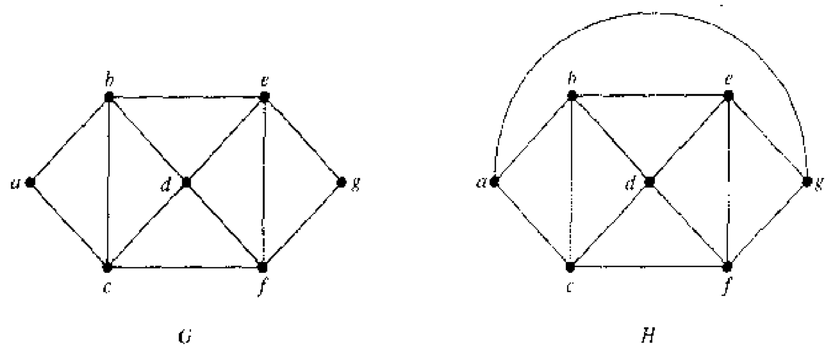


FIGURE 3 The Simple Graphs *G* and *H*.

such a coloring. Second, we must show that the graph cannot be colored using fewer than n colors. The following examples illustrate how chromatic numbers can be found.

EXAMPLE 1 What are the chromatic numbers of the graphs G and H shown in Figure 3?

Solution: The chromatic number of G is at least 3, since the vertices a, b, and c must be assigned different colors. To see if G can be colored with three colors, assign red to a, blue to b, and green to c. Then, d can (and must) be colored red since it is adjacent to b and c. Furthermore, e can (and must) be colored green since it is adjacent only to vertices colored red and blue, and f can (and must) be colored blue since it is adjacent only to vertices colored red and green. Finally, g can (and must) be colored red since it is adjacent only to vertices colored blue and green. This produces a coloring of G using exactly three colors. Figure 4 displays such a coloring.

The graph H is made up of the graph G with an edge connecting a and g. Any attempt to color H using three colors must follow the same reasoning as that used to color G, except at the last stage, when all vertices other than g have been colored. Then, since g is adjacent (in H) to vertices colored red, blue, and green, a fourth color, say brown, needs to be used. Hence, H has a chromatic number equal to 4. A coloring of H is shown in Figure 4. ■

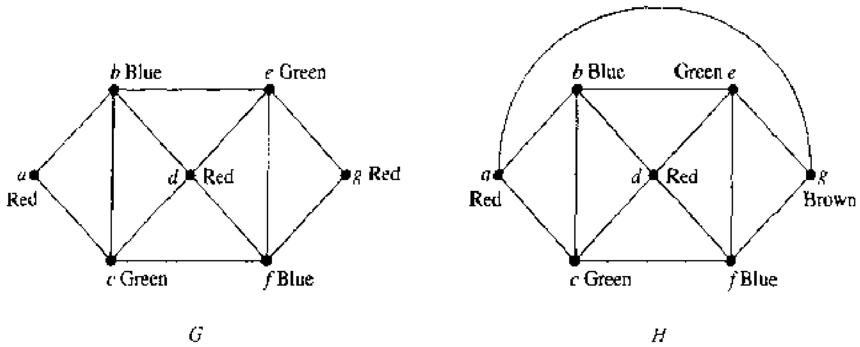


FIGURE 4 Colorings of the Graphs *G* and *H*.

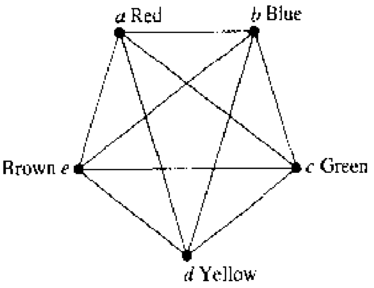


FIGURE 5 A Coloring of K_5.

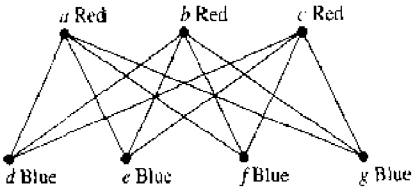


FIGURE 6 A Coloring of $K_{3,4}$.

EXAMPLE 2 What is the chromatic number of K_n?

Solution: A coloring of K_n can be constructed using n colors by assigning a different color to each vertex. Is there a coloring using fewer colors? The answer is no. No two vertices can be assigned the same color, since every two vertices of this graph are adjacent. Hence, the chromatic number of $K_n = n$. (Recall that K_n is not planar when $n \geq 5$, so this result does not contradict the four color theorem.) A coloring of K_5 using five colors is shown in Figure 5. ■

EXAMPLE 3 What is the chromatic number of the complete bipartite graph $K_{m,n}$, where m and n are positive integers?

Solution: The number of colors needed may seem to depend on m and n. However, only two colors are needed. Color the set of m vertices with one color and the set of n vertices with a second color. Since edges connect only a vertex from the set of m vertices and a vertex from the set of n vertices, no two adjacent vertices have the same color. A coloring of $K_{3,4}$ with two colors is displayed in Figure 6. ■

Every connected bipartite simple graph has a chromatic number of 2, or 1, since the reasoning used in Example 3 applies to any such graph. Conversely, every graph with a chromatic number of 2 is bipartite. (See Exercises 23 and 24 at the end of this section.)

EXAMPLE 4 What is the chromatic number of the graph C_n? (Recall that C_n is the cycle with n vertices.)

Solution: We will first consider some individual cases. To begin, let $n = 6$. Pick a vertex and color it red. Proceed clockwise in the planar depiction of C_6 shown in Figure 7. It is necessary to assign a second color, say blue, to the next vertex reached. Continue in the clockwise direction; the third vertex can be colored red, the fourth vertex blue, and the fifth vertex red. Finally, the sixth vertex, which is adjacent to the first, can be colored blue. Hence, the chromatic number of C_6 is 2. Figure 7 displays the coloring constructed here.

Next, let $n = 5$ and consider C_5. Pick a vertex and color it red. Proceeding clockwise, it is necessary to assign a second color, say blue, to the next vertex reached.

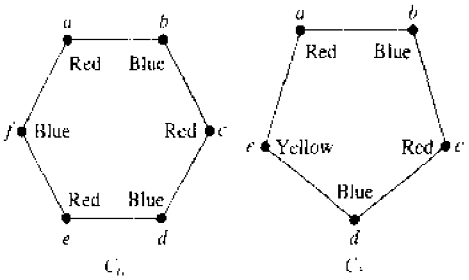


FIGURE 7 **Colorings of C_5 and C_6.**

Continuing in the clockwise direction, the third vertex can be colored red, and the fourth vertex can be colored blue. The fifth vertex cannot be colored either red or blue, since it is adjacent to the fourth vertex and the first vertex. Consequently, a third color is required for this vertex. Note that we would have also needed three colors if we had colored vertices in the counterclockwise direction. Thus, the chromatic number of C_5 is 3. A coloring of C_5 using three colors is displayed in Figure 7.

In general, two colors are needed to color C_n when n is even. To construct such a coloring, simply pick a vertex and color it red. Then proceed around the graph in a clockwise direction (using a planar representation of the graph) coloring the second vertex blue, the third vertex red, and so on. The nth vertex can be colored blue, since the two vertices adjacent to it, namely the $(n-1)$st and the first vertices, are both colored red.

When n is odd and $n > 1$, the chromatic number of C_n is 3. To see this, pick an initial vertex. To use only two colors, it is necessary to alternate colors as the graph is traversed in a clockwise direction. However, the nth vertex reached is adjacent to two vertices of different colors, namely, the first and $(n-1)$st. Hence, a third color must be used. ◼

web The best algorithms known for finding the chromatic number of a graph have exponential worst-case time complexity (in the number of vertices of the graph). Even the problem of finding an approximation to the chromatic number of a graph is difficult. It has been shown that if there were an algorithm with polynomial worst-case time complexity which could approximate the chromatic number of a graph up to a factor of 2 (that is, construct a bound which was no more than double the chromatic number of the graph), then an algorithm with polynomial worst-case time complexity for finding the chromatic number of the graph would also exist.

APPLICATIONS OF GRAPH COLORINGS

Graph coloring has a variety of applications to problems involving scheduling and assignments. Examples of such applications will be given here. The first application deals with the scheduling of final exams.

EXAMPLE 5 Scheduling Final Exams How can the final exams at a university be scheduled so that no student has two exams at the same time?

Solution: This scheduling problem can be solved using a graph model, with vertices representing courses and with an edge between two vertices if there is a common student

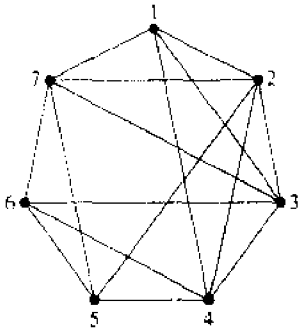


FIGURE 8 **The Graph Representing the Scheduling of Final Exams.**

in the courses they represent. Each time slot for a final exam is represented by a different color. A scheduling of the exams corresponds to a coloring of the associated graph.

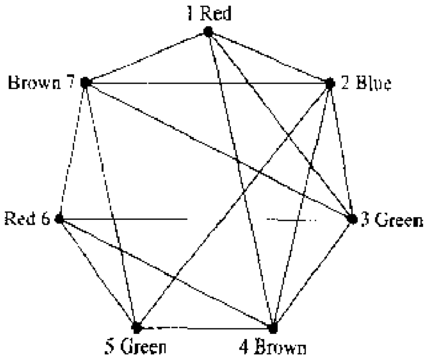
For instance, suppose there are seven finals to be scheduled. Suppose the courses are numbered 1 through 7. Suppose that the following pairs of courses have common students: 1 and 2, 1 and 3, 1 and 4, 1 and 7, 2 and 3, 2 and 4, 2 and 5, 2 and 7, 3 and 4, 3 and 6, 3 and 7, 4 and 5, 4 and 6, 5 and 6, 5 and 7, and 6 and 7. In Figure 8 the graph associated with this set of classes is shown. A scheduling consists of a coloring of this graph.

Since the chromatic number of this graph is 4 (the reader should verify this), four time slots are needed. A coloring of the graph using four colors and the associated schedule are shown in Figure 9. ■

Now consider an application to the assignment of television channels.

EXAMPLE 6 Frequency Assignments Television channels 2 through 13 are assigned to stations in North America so that no two stations within 150 miles can operate on the same channel. How can the assignment of channels be modeled by graph coloring?

Solution: Construct a graph by assigning a vertex to each station. Two vertices are connected by an edge if they are located within 150 miles of each other. An assignment of channels corresponds to a coloring of the graph, where each color represents a different channel. ■



Time Period	Courses
I	1, 6
II	2
III	3, 5
IV	4, 7

FIGURE 9 **Using a Coloring to Schedule Final Exams.**

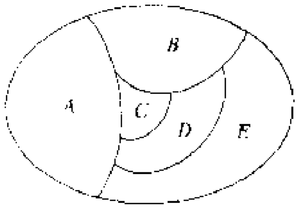
An application of graph coloring to compilers follows.

EXAMPLE 7 Index Registers In efficient compilers the execution of loops is speeded up when frequently used variables are stored temporarily in index registers in the central processing unit, instead of in regular memory. For a given loop, how many index registers are needed? This problem can be addressed using a graph coloring model. To set up the model, let each vertex of a graph represent a variable in the loop. There is an edge between two vertices if the variables they represent must be stored in index registers at the same time during the execution of the loop. Thus, the chromatic number of the graph gives the number of index registers needed, since different registers must be assigned to variables when the vertices representing these variables are adjacent in the graph.■

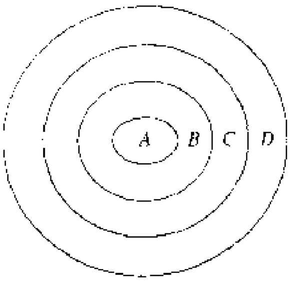
Exercises

1. Construct the dual graphs for each of the following maps.

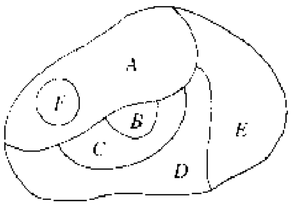
a)



b)



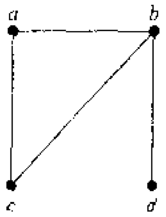
c)



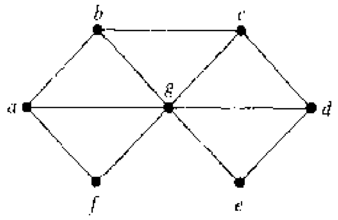
2. Find the number of colors needed to color the maps in Exercise 1 so that no two adjacent regions have the same color.

In Exercises 3–9 find the chromatic number of the given graph.

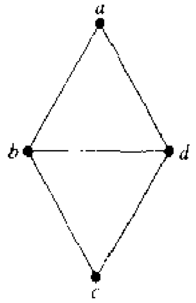
3.



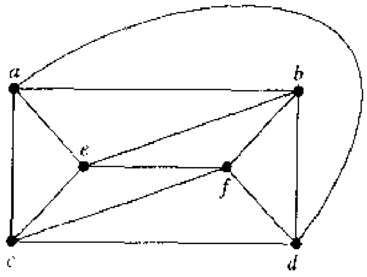
4.



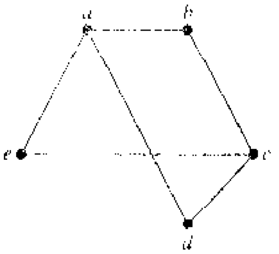
5.



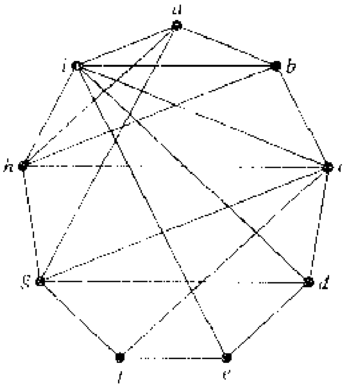
6.



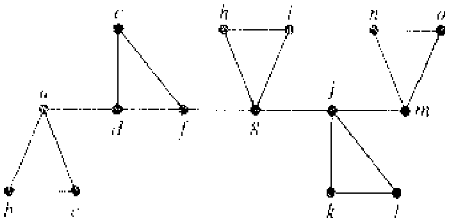
7.



8.



9.



10. For the graphs in Exercises 3–9, decide whether it is possible to decrease the chromatic number by removing a single vertex and all edges incident with it.

11. Which graphs have a chromatic number of 1?

12. What is the least number of colors needed to color a map of the United States? Do not consider adjacent states that meet only at a corner. Suppose that Michigan is one region. Consider the vertices representing Alaska and Hawaii as isolated vertices.

13. What is the chromatic number of W_n?

14. Show that a simple graph which has a circuit with an odd number of vertices in it cannot be colored using two colors.

15. Schedule the final exams for Math 115, Math 116, Math 185, Math 195, CS 101, CS 102, CS 273, and CS 473, using the fewest number of different time slots, if there are no students taking both Math 115 and CS 473, both Math 116 and CS 473, both Math 195 and CS 101, both Math 195 and CS 102, both Math 115 and Math 116, both Math 115 and Math 185, and both Math 185 and Math 195, but there are students in every other combination of courses.

16. How many different channels are needed for six stations located at the distances shown in the table, if two stations cannot use the same channel when they are within 150 miles of each other?

	1	*2*	*3*	*4*	*5*	*6*
1	—	85	175	200	50	100
2	85	—	125	175	100	160
3	175	125	—	100	200	250
4	200	175	100	—	210	220
5	50	100	200	210	—	100
6	100	160	250	220	100	—

17. The mathematics department has six committees that meet once a month. How many different meeting times must be used to ensure that no one is scheduled to be at two meetings at the same time if the committees are C_1 = {*Arlinghaus, Brand, Zaslavsky*}, C_2 = {*Brand, Lee, Rosen*}, C_3 = {*Arlinghaus, Rosen, Zaslavsky*}, C_4 = {*Lee, Rosen, Zaslavsky*}, C_5 = {*Arlinghaus, Brand*}, and C_6 = {*Brand, Rosen, Zaslavsky*}.

18. A zoo wants to set up natural habitats in which to exhibit its animals. Unfortunately, some animals will eat some of the others when given the opportunity. How can a graph model and a coloring be used to determine the number of different habitats needed and the placement of the animals in these habitats?

An **edge coloring** of a graph is an assignment of colors to edges so that edges incident with a common vertex are assigned different colors. The **edge chromatic number** of a graph is the smallest number of colors that can be used in an edge coloring of the graph.

19. Find the edge chromatic numbers of each of the graphs in Exercises 3–9.

***20.** Find the edge chromatic numbers of

a) K_n. b) $K_{m,n}$. c) C_n. d) W_n.

21. Seven variables occur in a loop of a computer program. The variables and the steps during which they must be stored are: *t*: steps 1 through 6; *u*: step 2; *v*: steps 2 through 4; *w*: steps 1, 3, and 5; *x*: steps 1 and 6; *y*: steps 3 through 6; and *z*: steps 4 and 5. How many different index registers are needed to store these variables during execution?

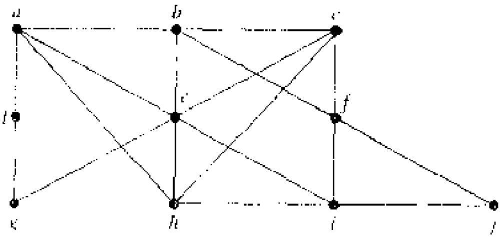
22. What can be said about the chromatic number of a graph that has K_n as a subgraph?

23. Show that a simple graph with a chromatic number of 2 is bipartite.

24. Show that a connected bipartite graph has a chromatic number of 2.

The following algorithm can be used to color a simple graph. First, list the vertices $v_1, v_2, v_3, \ldots, v_n$ in order of decreasing degree so that $\deg(v_1) \geq \deg(v_2) \geq \cdots \geq \deg(v_n)$. Assign color 1 to v_1 and to the next vertex in the list not adjacent to v_1 (if one exists), and successively to each vertex in the list not adjacent to a vertex already assigned color 1. Then assign color 2 to the first vertex in the list not already colored. Successively assign color 2 to vertices in the list that have not been already colored and are not adjacent to vertices assigned color 2. If uncolored vertices remain, assign color 3 to the first vertex in the list not yet colored, and use color 3 to successively color those vertices not already colored and not adjacent to vertices assigned color 3. Continue this process until all vertices are colored.

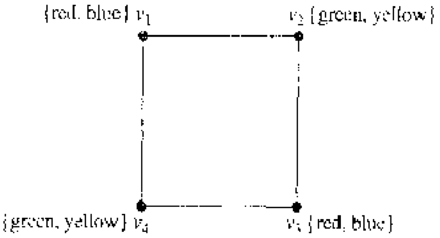
25. Construct a coloring of the following graph using this algorithm.



***26.** Use pseudocode to describe this coloring algorithm.

***27.** Show that the coloring produced in this algorithm may use more colors than are necessary to color a graph.

A **k-tuple coloring** of a graph G is an assignment of a set of k different colors to the vertices of G so that no two adjacent vertices are assigned a common color. We denote by $\chi_k(G)$ the smallest positive integer n such that G has a k-tuple coloring using n colors. For example, $\chi_2(C_4) = 4$. To see this, note that using only four colors we can assign two colors to each vertex of C_4, as illustrated, so that no two adjacent vertices are assigned the same color. Furthermore, no fewer than four colors suffice because the vertices v_1 and v_2 each must be assigned two colors, and a common color cannot be assigned to both v_1 and v_2. (For more information about k-tuple coloring, see [MiRo90].)



28. Find the following values:

a) $\chi_2(K_3)$ **b)** $\chi_2(K_4)$
c) $\chi_2(W_4)$ **d)** $\chi_2(C_5)$
e) $\chi_2(K_{3,4})$ **f)** $\chi_3(K_5)$
***g)** $\chi_3(C_5)$ **h)** $\chi_3(K_{4,5})$

***29.** Let G and H be the graphs displayed in Figure 3. Find

a) $\chi_2(G)$. **b)** $\chi_2(H)$. **c)** $\chi_3(G)$. **d)** $\chi_3(H)$.

30. What is $\chi_k(G)$ if G is a bipartite graph and k is a positive integer?

31. Frequencies for mobile radio (or cellular) telephones are assigned by zones. Each zone is assigned a set of frequencies to be used by vehicles in that zone. The same frequency cannot be used in zones where interference is a problem. Explain how a k-tuple coloring can be used to assign k frequencies to each mobile radio zone in a region.

Key Terms and Results

TERMS

undirected edge: an edge associated to a set $\{u, v\}$, where u and v are vertices

directed edge: an edge associated to an ordered pair (u, v), where u and v are vertices

multiple edges: distinct edges connecting the same vertices

loop: an edge connecting a vertex with itself

undirected graph: a set of vertices together with a set of undirected edges that connect these vertices

simple graph: an undirected graph with no multiple edges or loops

multigraph: an undirected graph that may contain multiple edges but no loops

pseudograph: an undirected graph that may contain multiple edges and loops

directed graph: a set of vertices together with a set of directed edges that connect these vertices

directed multigraph: a graph with directed edges that may contain multiple directed edges

adjacent: two vertices are adjacent if there is an edge between them

incident: an edge is incident with a vertex if the vertex is an endpoint of that edge

deg(v) (degree of the vertex v in an undirected graph): the number of edges incident with v

$\deg^-(v)$ (the in-degree of the vertex v in a graph with directed edges): the number of edges with v as their terminal vertex

$\deg^+(v)$ (the out-degree of the vertex v in a graph with directed edges): the number of edges with v as their initial vertex

underlying undirected graph of a graph with directed edges: the undirected graph obtained by ignoring the directions of the edges

K_n (complete graph on n vertices): the undirected graph with n vertices where each pair of vertices is connected by an edge

bipartite graph: a graph with vertex set partitioned into subsets V_1 and V_2 such that each edge connects a vertex in V_1 and a vertex in V_2

$K_{m,n}$ (complete bipartite graph): the graph with vertex set partitioned into a subset of m elements and a subset of n elements such that two vertices are connected by an edge if and only if one is in the first subset and the other is in the second subset

C_n (cycle of size n), $n \geq 3$: the graph with n vertices $v_1, v_2, \ldots, v_n$ and edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}$

W_n (wheel of size n), $n \geq 3$: the graph obtained from C_n by adding a vertex and edges from this vertex to the original vertices in C_n

Q_n (n-cube), $n \geq 1$: the graph that has the 2^n bit strings of length n as its vertices and edges connecting every pair of bit strings that differ by exactly one bit

isolated vertex: a vertex of degree 0

pendant vertex: a vertex of degree 1

regular graph: a graph where all vertices have the same degree

subgraph of a graph $G = (V, E)$: a graph (W, F) where W is a subset of V and F is a subset of E

$G_1 \cup G_2$ (union of G_1 and G_2): the graph $(V_1 \cup V_2, E_1 \cup E_2)$ where $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$

adjacency matrix: a matrix representing a graph using the adjacency of vertices

incidence matrix: a matrix representing a graph using the incidence of edges and vertices

isomorphic simple graphs: the simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are isomorphic if there is a one-to-one correspondence f from V_1 to V_2 such that $\{f(v_1), f(v_2)\} \in E_2$ if and only if $\{v_1, v_2\} \in E_1$ for all v_1 and v_2 in V_1

invariant: a property that isomorphic graphs either both have or both do not have

path from u to v in an undirected graph: a sequence of one or more edges $e_1, e_2, \ldots, e_n$ where e_i is associated to $\{x_i, x_{i-1}\}$ for $i = 0, 1, \ldots, n$ where $x_0 = u$ and $x_1 = v$

path from u to v in a graph with directed edges: a sequence of one or more edges $e_1, e_2, \ldots, e_n$ where e_i is associated to (x_i, x_{i-1}) for $i = 0, 1, \ldots, n$ where $x_0 = u$ and $x_{n+1} = v$

simple path: a path that does not contain an edge more than once

circuit: a path that begins and ends at the same vertex

connected graph: an undirected graph with the property that there is a path between every pair of vertices in the graph

connected components: the set of connected subgraphs of a graph such that no two of these subgraphs have a vertex in common

Euler circuit: a circuit that contains every edge of a graph exactly once

Euler path: a path that contains every edge of a graph exactly once

Hamilton path: a path $x_0, x_1, \ldots, x_n$ in a simple graph $G = (V, E)$ such that $\{x_0, x_1, \ldots, x_n\} = V$ and $x_i \neq x_j$ for $0 \leq i < j \leq n$

Hamilton circuit: a circuit $x_0, x_1, \ldots, x_n, x_0$ in a simple graph such that $x_0, x_1, \ldots, x_n$ is a Hamilton path

weighted graph: a graph with numbers assigned to its edges

shortest path problem: the problem of determining the path in a weighted graph such that the sum of the weights of the edges in this path is a minimum over all paths between specified vertices

traveling salesman problem: the problem that asks for the circuit of shortest total length which visits every vertex of the graph exactly once

planar graph: a graph that can be drawn in the plane with no crossings

regions of a representation of a planar graph: the regions the plane is divided into by the planar representation of the graph

elementary subdivision: the removal of an edge $\{u, v\}$ of an undirected graph and the addition of a new vertex w together with edges $\{u, w\}$ and $\{w, v\}$

homeomorphic: two undirected graphs are homeomorphic if they can be obtained from the same graph by a sequence of elementary subdivisions

graph coloring: an assignment of colors to the vertices of a graph so that no two adjacent vertices have the same color

chromatic number: the minimum number of colors needed in a coloring of a graph

RESULTS

There is an Euler circuit in a connected multigraph if and only if every vertex has even degree.

There is an Euler path in a connected multigraph if and only if at most two vertices have odd degree.

Dijkstra's algorithm: a procedure for finding the shortest path between two vertices in a weighted graph (see page 494).

Euler's formula: $r = e - v + 2$ where r, e, and v are the number of regions of a planar representation, the number of edges, and the number of vertices, respectively, of a planar graph.

Kuratowski's theorem: A graph is nonplanar if and only if it contains a subgraph homeomorphic to $K_{3,3}$ or K_5. (Proof beyond scope of this book.)

The four color theorem: Every planar graph can be colored using no more than four colors. (Proof far beyond the scope of this book!)

Review Questions

1. a) Define a simple graph, a multigraph, a pseudograph, a directed graph, and a directed multigraph.
   b) Use an example to show how each of the types of graph in part (a) can be used in modeling. For example, explain how to model different aspects of a computer network or airline routes.
2. Give at least four examples of how graphs are used in modeling.
3. What is the relationship between the sum of the degrees of the vertices in an undirected graph and the number of edges in this graph? Explain why this relationship holds.
4. Why must there be an even number of vertices of odd degree in an undirected graph?
5. What is the relationship between the sum of the in-degrees and the sum of the out-degrees of the vertices in a directed graph? Explain why this relationship holds.
6. Describe the following families of graphs.
   a) K_n, the complete graph on n vertices
   b) $K_{m,n}$, the complete bipartite graph on m and n vertices
   c) C_n, the cycle with n vertices
   d) W_n, the wheel of size n
   e) Q_n, the n-cube
7. How many vertices and how many edges are there in each of the graphs in the families in Question 6?
8. a) What is a bipartite graph?
   b) Which of the graphs K_n, C_n, and W_n are bipartite?
   c) How can you determine whether an undirected graph is bipartite?
9. a) Describe three different methods that can be used to represent a graph.
   b) Draw a simple graph with at least five vertices and eight edges. Illustrate how it can be represented using the methods you described in part (a).
10. a) What does it mean for two simple graphs to be isomorphic?
    b) What is meant by an invariant with respect to isomorphism for simple graphs? Give at least five examples of such invariants.
    c) Give an example of two graphs that have the same numbers of vertices, edges, and degrees of vertices, but that are not isomorphic.
    d) Is a set of invariants known that can be used to efficiently determine whether two simple graphs are isomorphic?
11. a) What does it mean for a graph to be connected?
    b) What are the connected components of a graph?
12. a) Explain how an adjacency matrix can be used to represent a graph.
    b) How can adjacency matrices be used to determine whether a function from the vertex set of a graph G to the vertex set of a graph H is an isomorphism?
    c) How can the adjacency matrix of a graph be used to determine the number of paths of length r, where r is a positive integer, between two vertices of a graph?
13. a) Define an Euler circuit and an Euler path in an undirected graph.
    b) Describe the famous Königsberg bridge problem and explain how to rephrase it in terms of an Euler circuit.
    c) How can it be determined whether an undirected graph has an Euler path?
    d) How can it be determined whether an undirected graph has an Euler circuit?
14. a) Define a Hamilton circuit in a simple graph.
    b) Give some properties of a simple graph that imply that it does not have a Hamilton circuit.
15. Give examples of at least two problems that can be solved by finding the shortest path in a weighted graph.
16. a) Describe Dijkstra's algorithm for finding the shortest path in a weighted graph between two vertices.
    b) Draw a weighted graph with at least 10 vertices and 20 edges. Use Dijkstra's algorithm to find the shortest path between two vertices of your choice in the graph.
17. a) What does it mean for a graph to be planar?
    b) Give an example of a nonplanar graph.
18. a) What is Euler's formula for planar graphs?
    b) How can Euler's formula for planar graphs be used to show that a simple graph is non-planar?
19. State Kuratowski's theorem on the planarity of graphs and explain how it characterizes which graphs are planar.
20. a) Define the chromatic number of a graph.
    b) What is the chromatic number of the graph K_n when n is a positive integer?

c) What is the chromatic number of the graph C_n when n is a positive integer greater than 2?

d) What is the chromatic number of the graph $K_{m,n}$ when m and n are positive integers?

21. State the four color theorem. Are there graphs that cannot be colored with four colors?

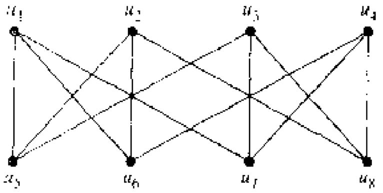
22. Explain how graph coloring can be used in modeling. Use at least two different examples.

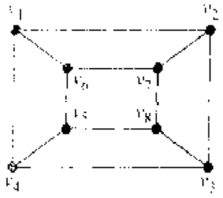
Supplementary Exercises

1. How many edges does a 50-regular graph with 100 vertices have?
2. How many nonisomorphic subgraphs does K_3 have?

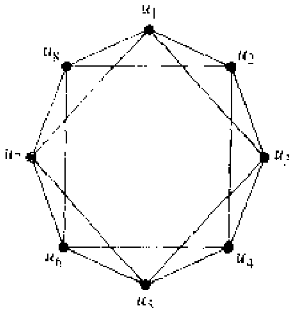
In Exercises 3-5 determine whether the given pair of graphs is isomorphic.

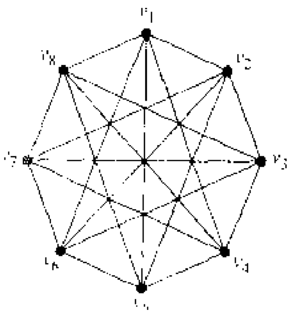
3.



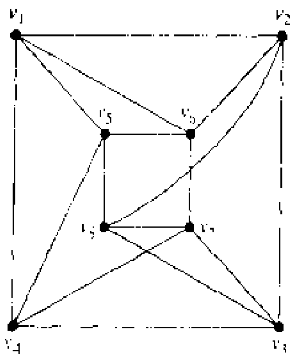


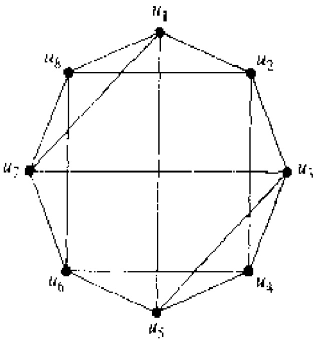
4.





*5.





The **complete m-partite graph** $K_{n_1,n_2,\ldots,n_m}$ has vertices partitioned into m subsets of $n_1, n_2, \ldots, n_m$ elements each, and vertices are adjacent if and only if they are in different subsets in the partition.

6. Draw the following graphs.

a) $K_{1,2,3}$ **b)** $K_{2,2,2}$ **c)** $K_{1,2,2,3}$

***7.** How many vertices and how many edges does the complete m-partite graph $K_{n_1,n_2,\ldots,n_m}$ have?

***8.** a) Prove or disprove that there are always two vertices with the same degree in a finite simple graph having at least two vertices.

b) Do the same as in part (a) for finite multigraphs.

Let $G = (V, E)$ be a simple graph. The **subgraph induced** by a subset W of the vertex set V is the graph (W, F), where the edge set F contains an edge in E if and only if both endpoints of this edge are in W.

9. Consider the graph shown in Figure 3 of Section 7.4. Find the subgraphs induced by

a) $\{a, b, c\}$.

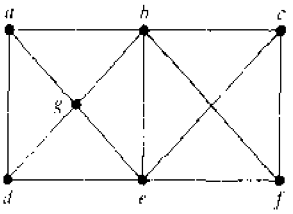
b) $\{a, e, g\}$.

c) $\{b, c, f, g, h\}$.

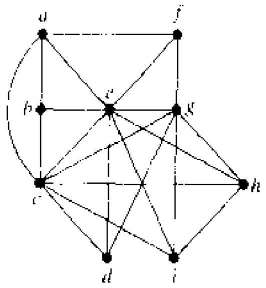
10. Let n be a positive integer. Show that a subgraph induced by a nonempty subset of the vertex set of K_n is a complete graph.

web A **clique** in a simple undirected graph is a complete subgraph that is not contained in any larger complete subgraph. In Exercises 11–13 find all cliques in the given graph.

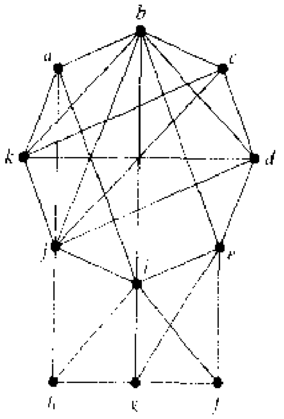
11.



12.

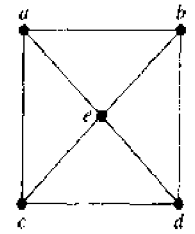


13.

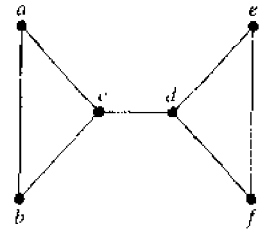


A **dominating set** of vertices in a simple graph is a set of vertices such that every other vertex is adjacent to at least one vertex of this set. A dominating set with the least number of vertices is called a **minimum dominating set**. In Exercises 14–16 find a minimum dominating set for the given graph.

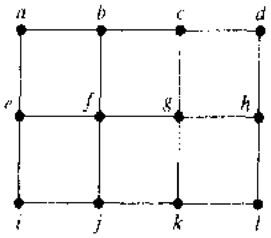
14.



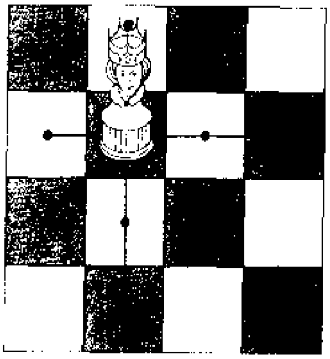
15.



16.



A simple graph can be used to determine the minimum number of queens on a chessboard that control the entire chessboard. An $n \times n$ chessboard has n^2 squares in an $n \times n$ configuration. A queen in a given position controls all squares in the same row, the same column, and on the two diagonals containing this square, as illustrated. The appropriate simple graph has n^2 vertices, one for each square, and two vertices are adjacent if a queen in the square represented by one of the vertices controls the square represented by the other vertex.



The Squares Controlled by a Queen

17. Construct the simple graph representing the $n \times n$ chessboard with edges representing the control of squares by queens for

a) $n = 3$. **b)** $n = 4$.

18. Explain how the concept of a minimum dominating set applies to the problem of determining the minimum number of queens controlling an $n \times n$ chessboard.

****19.** Find the minimum number of queens controlling an $n \times n$ chessboard for

a) $n = 3$. **b)** $n = 4$. **c)** $n = 5$.

20. Suppose that G_1 and H_1 are isomorphic and that G_2 and H_2 are isomorphic. Prove or disprove that $G_1 \cup G_2$ and $H_1 \cup H_2$ are isomorphic.

21. Show that each of the following properties is an invariant that isomorphic simple graphs either both have or both do not have.

a) connectedness
b) the existence of a Hamilton circuit
c) the existence of an Euler circuit
d) having crossing number C
e) having n isolated vertices
f) being bipartite

22. How can the adjacency matrix of $\overline{G}$ be found from the adjacency matrix of G, where G is a simple graph?

23. How many nonisomorphic connected bipartite simple graphs are there with four vertices?

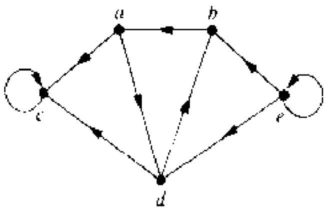
***24.** How many nonisomorphic simple connected graphs with five vertices are there

a) with no vertex of degree more than 2?
b) with chromatic number equal to 4?
c) that are nonplanar?

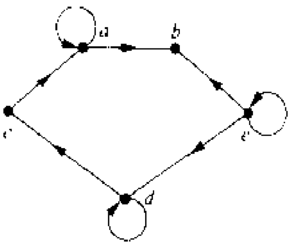
A directed graph is **self-converse** if it is isomorphic to its converse.

25. Determine whether the following graphs are self-converse.

a)



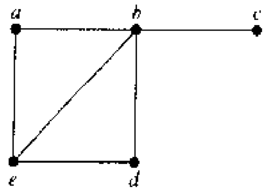
b)



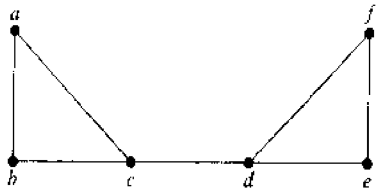
26. Show that if the directed graph G is self-converse and H is a directed graph isomorphic to G, then H is also self-converse.

An **orientation** of an undirected simple graph is an assignment of directions to its edges so that the resulting directed graph is strongly connected. When an orientation of an undirected graph exists, this graph is called **orientable**. In Exercises 27–29 determine whether the given simple graph is orientable.

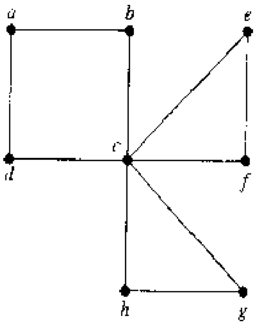
27.



28.



29.



30. Because traffic is growing heavy in the central part of a city, traffic engineers are planning to change all the streets, which are currently two-way, into one-way streets. Explain how to model this problem.

***31.** Show that a graph is not orientable if it has a cut edge.

A **tournament** is a simple directed graph such that if u and v are distinct vertices in the graph, exactly one of (u, v) and (v, u) is an edge of the graph.

32. How many different tournaments are there with n vertices?

33. What is the sum of the in-degree and out-degree of a vertex in a tournament?

***34.** Show that every tournament has a Hamilton path.

35. Given two chickens in a flock, one of them is dominant. This defines the **pecking order** of the flock. How can a tournament be used to model pecking order?

36. Suppose that G is a connected multigraph with $2k$ vertices of odd degree. Show that there exist k subgraphs that have G as their union, where each of these subgraphs has an Euler path and where no two of these subgraphs have an edge in common. (*Hint:* Add k edges to the graph connecting pairs of vertices of odd degree and use an Euler circuit in this larger graph.)

***37.** Let G be a simple graph with n vertices. The **bandwidth** of G, denoted by $B(G)$, is the minimum, over all permutations, $a_1, a_2, \ldots, a_n$ of the vertices of G, of $\max\{|i - j| \mid a_i$ and a_j are adjacent$\}$. That is, the bandwidth is the minimum over all listings of the vertices of the maximum difference in the indices assigned to adjacent vertices. Find the bandwidths of the following graphs.

a) K_5 **b)** $K_{1,3}$ **c)** $K_{2,3}$
d) $K_{3,3}$ **e)** Q_3 **f)** C_5

***38.** The **distance** between two distinct vertices v_1 and v_2 of a connected simple graph is the length (number of edges) of the shortest path between v_1 and v_2. The **radius** of a graph is the minimum over all vertices v of the maximum distance from v to another vertex. The **diameter** of a graph is the maximum distance between two distinct vertices. Find the radius and diameter of

a) K_6. **b)** $K_{4,5}$. **c)** Q_3. **d)** C_6.

***39.** **a)** Show that if the diameter of the simple graph G is at least 4, then the diameter of its complement $\overline{G}$ is no more than 2.

b) Show that if the diameter of the simple graph G is at least 3, then the diameter of its complement $\overline{G}$ is no more than 3.

***40.** Suppose that a multigraph has $2m$ vertices of odd degree. Show that any circuit that contains every edge of the graph must contain at least m edges more than once.

41. Find the second shortest path between the vertices a and z in Figure 3 of Section 7.6.

42. Devise an algorithm for finding the second shortest path between two vertices in a simple connected weighted graph.

43. Find the shortest path between the vertices a and z that passes through the vertex e in the weighted graph in Figure 4 in Section 7.6.

44. Devise an algorithm for finding the shortest path between two vertices in a simple connected weighted graph that passes through a specified third vertex.

***45.** Show that if G is a simple graph with at least 11 vertices, then either G or $\overline{G}$, the complement of G, is nonplanar.

web A set of vertices in a graph is called **independent** if no two vertices in the set are adjacent. The **independence number** of a graph is the maximum number of vertices in an independent set of vertices for the graph.

***46.** What is the independence number of

a) K_n? **b)** C_n? **c)** Q_n? **d)** $K_{m,n}$?

47. Show that the number of vertices in a simple graph is less than or equal to the product of the independence number and the chromatic number of the graph.

48. Show that the chromatic number of a graph is less than or equal to $v - i + 1$, where v is the number of vertices in the graph and i is the independence number of this graph.

49. Suppose that to generate a random simple graph with n vertices we first choose a real number p with $0 \leq p \leq 1$. For each of the $C(n, 2)$ pairs of distinct vertices we generate a random number x between 0 and 1. If $0 \leq x \leq p$, we connect these two vertices with an edge; otherwise these vertices are not connected.

a) What is the probability that a graph with m edges where $0 \leq m \leq C(n, 2)$ is generated?

b) What is the expected number of edges in a randomly generated graph with n vertices if each edge is included with probability p?

c) Show that if $p = 1/2$ then every simple graph with n vertices is equally likely to be generated.

A property retained whenever additional edges are added to a simple graph (without adding vertices) is called **monotone increasing,** and a property that is retained whenever edges are removed from a simple graph (without removing vertices) is called **monotone decreasing.**

50. For each of the following properties, determine whether it is monotone increasing and determine whether it is monotone decreasing.

a) The graph G is connected.
b) The graph G is not connected.
c) The graph G has an Euler circuit.
d) The graph G has a Hamilton circuit.
e) The graph G is planar.
f) The graph G has chromatic number four.
g) The graph G has radius three.
h) The graph G has diameter three.

51. Show that the graph property P is monotone increasing if and only if the graph property Q is monotone decreasing where Q is the property of not having property P.

****52.** Suppose that P is a monotone increasing property of simple graphs. Show that the probability a random graph with n vertices has property P is a monotonic nondecreasing function of p, the probability an edge is chosen to be in the graph.

Computer Projects

WRITE PROGRAMS WITH THE FOLLOWING INPUT AND OUTPUT.

1. Given the vertex pairs associated to the edges of an undirected graph, determine the degree of each vertex.

2. Given the ordered pairs of vertices associated to the edges of a directed graph, determine the in-degree and out-degree of each vertex.

3. Given the list of edges of a simple graph, determine whether the graph is bipartite.
4. Given the vertex pairs associated to the edges of a graph, construct an adjacency matrix for the graph. (Produce a version that works when loops, multiple edges, or directed edges are present.)
5. Given an adjacency matrix of a graph, list the edges of this graph and give the number of times each edge appears.
6. Given the vertex pairs associated to the edges of an undirected graph and the number of times each edge appears, construct an incidence matrix for the graph.
7. Given an incidence matrix of an undirected graph, list its edges and give the number of times each edge appears.
8. Given a positive integer n, generate an undirected graph by producing an adjacency matrix for the graph so that all simple graphs are equally likely to be generated.
9. Given a positive integer n, generate a directed graph by producing an adjacency matrix for the graph so that all directed graphs are equally likely to be generated.
10. Given the lists of edges of two simple graphs with no more than six vertices, determine whether the graphs are isomorphic.
11. Given an adjacency matrix of a graph and a positive integer n, find the number of paths of length n between two vertices. (Produce a version that works for directed and undirected graphs.)

*12. Given the list of edges of a simple graph, determine whether it is connected and find the number of connected components if it is not connected.

13. Given the vertex pairs associated to the edges of a multigraph, determine whether it has an Euler circuit and, if not, whether it has an Euler path. Construct an Euler path or circuit if it exists.

*14. Given the ordered pairs of vertices associated to the edges of a directed multigraph, construct an Euler path or Euler circuit, if such a path or circuit exists.

**15. Given the list of edges of a simple graph, produce a Hamilton circuit, or determine that the graph does not have such a circuit.

**16. Given the list of edges of a simple graph, produce a Hamilton path, or determine that the graph does not have such a path.

17. Given the list of edges and weights of these edges of a weighted connected simple graph and two vertices in this graph, find the length of the shortest path between them using Dijkstra's algorithm. Also, find this path.
18. Given the list of edges of an undirected graph, find a coloring of this graph using the algorithm given in the exercise set of Section 7.8.
19. Given a list of students and the courses that they are enrolled in, construct a schedule of final exams.
20. Given the distances between pairs of television stations, assign frequencies to these stations.

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. Display all the simple graphs with four vertices.
2. Display a full set of nonisomorphic simple graphs with six vertices.
3. Display a full set of nonisomorphic directed graphs with four vertices.
4. Generate at random 10 different simple graphs each with 20 vertices so that each such graph is equally likely to be generated.
5. Construct a Gray code where the code words are bit strings of length 6.
6. Construct knight's tours on chessboards of various sizes.
7. Determine whether each of the graphs you generated in Exercise 4 of this set is planar. If you can, determine the thickness of each of the graphs that are not planar.
8. Determine whether each of the graphs you generated in Exercise 4 of this set is connected. If a graph is not connected, determine the number of connected components of the graph.
9. Generate at random simple graphs with 10 vertices. Stop when you have constructed one with an Euler circuit. Display an Euler circuit in this graph.
10. Generate at random simple graphs with 10 vertices. Stop when you have constructed one with a Hamilton circuit. Display a Hamilton circuit in this graph.
11. Find the chromatic number of each of the graphs you generated in Exercise 4 of this set.

**12. Find the shortest path a traveling salesperson can take to visit each of the capitals of the 50 states in the U.S., traveling by air between cities in a straight line.

*13. Estimate the probability that a randomly generated simple graph with n vertices is connected for each positive integer n not exceeding 10 by generating a set of random simple graphs and determining whether each is connected.

**14. Work on the problem of determining whether the crossing number of $K(7, 7)$ is 77, 79, or 81. It is known that it equals one of these three values.

Writing Projects

RESPOND TO THE FOLLOWING QUESTIONS WITH ESSAYS USING OUTSIDE SOURCES.

1. Describe the origins and development of graph theory prior to the year 1900.
2. Discuss the applications of graph theory to the study of ecosystems.
3. Discuss the applications of graph theory to sociology and psychology.
4. Describe algorithms for drawing a graph on paper or on a display given the vertices and edges of the graph. What considerations arise in drawing a graph so that it has the best appearance for understanding its properties?
5. What are some of the capabilities that a software tool for inputting, displaying, and manipulating graphs should have? Which of these capabilities do available tools have?
6. Describe some of the algorithms available for determining whether two graphs are isomorphic and the computational complexity of these algorithms. What is the most efficient such algorithm currently known?
7. Define *de Bruijn sequences* and discuss how they arise in applications. Explain how de Bruijn sequences can be constructed using Euler circuits.
8. Describe the *Chinese Postman Problem* and explain how to solve this problem.
9. Describe some of the different conditions that imply that a graph has a Hamilton circuit.
10. Describe some of the strategies and algorithms used to solve the traveling salesman problem.
11. Describe several different algorithms for determining whether a graph is planar. What is the computational complexity of each of these algorithms?
12. In modeling, very large scale integration (VLSI) graphs are sometimes embedded in a book, with the vertices on the spine and the edges on pages. Define the *book number* of a graph and find the book number of various graphs including K_n for $n = 3, 4, 5,$ and 6.
13. Discuss the history of the four color theorem.
14. Describe the role computers played in the proof of the four color theorem. How can we be sure that a proof that relies on a computer is correct?
15. Describe and compare several different algorithms for coloring a graph, in terms of whether they produce a coloring with the least number of colors possible and in terms of their complexity.
16. Explain how graph multicolorings can be used in a variety of different models.
17. Explain how the theory of random graphs can be used in nonconstructive existence proofs of graphs with certain properties.

8

Trees

A connected graph that contains no simple circuits is called a tree. Trees were used as long ago as 1857, when the English mathematician Arthur Cayley used them to count certain types of chemical compounds. Since that time, trees have been employed to solve problems in a wide variety of disciplines, as the examples in this chapter will show.

Trees are particularly useful in computer science. For instance, trees are employed to construct efficient algorithms for locating items in a list. They are used to construct networks with the least expensive set of telephone lines linking distributed computers. Trees can be used to construct efficient codes for storing and transmitting data. Trees can model procedures that are carried out using a sequence of decisions. This makes trees valuable in the study of sorting algorithms.

8.1

Introduction to Trees

A genealogical chart of the Bernoullis, a famous family of Swiss mathematicians, is shown in Figure 1. Such a chart is also called a family tree. A family tree is a graph where the vertices represent family members and the edges represent parent–child relationships. The undirected graph that represents a genealogical chart is an example of a special type of graph called a **tree**.

DEFINITION 1. A *tree* is a connected undirected graph with no simple circuits.

Since a tree cannot have a simple circuit, a tree cannot contain multiple edges or loops. Therefore any tree must be a simple graph.

EXAMPLE 1 Which of the graphs shown in Figure 2 are trees?

Solution: G_1 and G_2 are trees, since both are connected graphs with no simple circuits. G_3 is not a tree because e, b, a, d, e is a simple circuit in this graph. Finally, G_4 is not a tree since it is not connected. ■

Any connected graph that contains no simple circuits is a tree. What about graphs containing no simple circuits that are not necessarily connected? These graphs are called

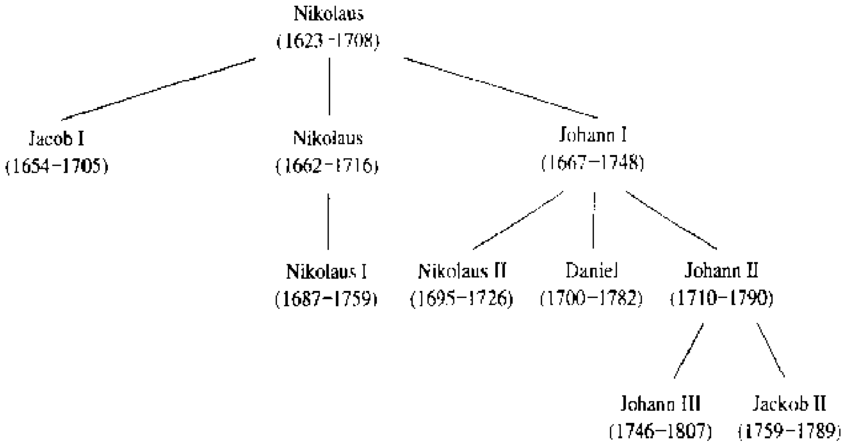

Nikolaus
(1623-1708)
Jacob I
(1654-1705)
Nikolaus
(1662-1716)
Johann I
(1667-1748)
Nikolaus I
(1687-1759)
Nikolaus II
(1695-1726)
Daniel
(1700-1782)
Johann II
(1710-1790)
Johann III
(1746-1807)
Jackob II
(1759-1789)

FIGURE 1 The Bernoulli Family of Mathematicians.

forests and have the property that each of their connected components is a tree. Figure 3 displays a forest.

Trees are often defined as undirected graphs with the property that there is a unique simple path between every pair of vertices. The following theorem shows that this alternative definition is equivalent to our definition.

THEOREM 1 **An undirected graph is a tree if and only if there is a unique simple path between any two of its vertices.**

Proof: First assume that T is a tree. Then T is a connected graph with no simple circuits. Let x and y be two vertices of T. Since T is connected, by Theorem 1 of Section 7.4 there is a simple path between x and y. Moreover, this path must be unique, for if there were a second such path, the path formed by combining the first path from x to y followed by the path from y to x obtained by reversing the order of the second path from x to y would form a circuit. This implies, using Exercise 35 of Section 7.4, that there is a

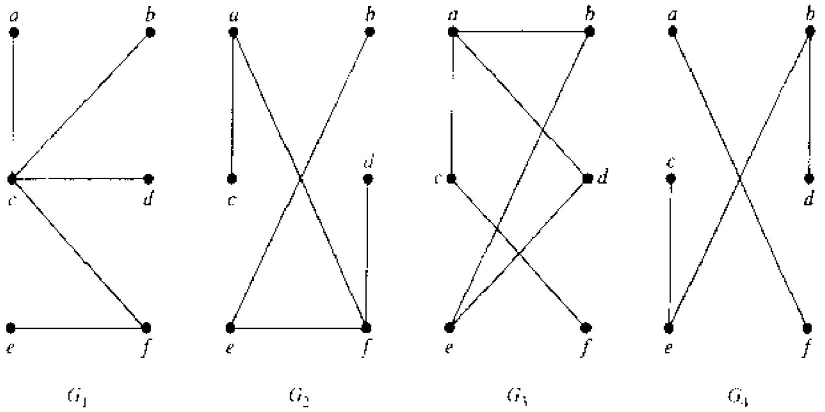

a
b
c
d
e
f
G_1
a
b
c
d
e
f
G_2
a
b
c
d
e
f
G_3
a
b
c
d
e
f
G_4

FIGURE 2 G_1 and G_2 Are Trees; G_3 and G_4 Are Not.

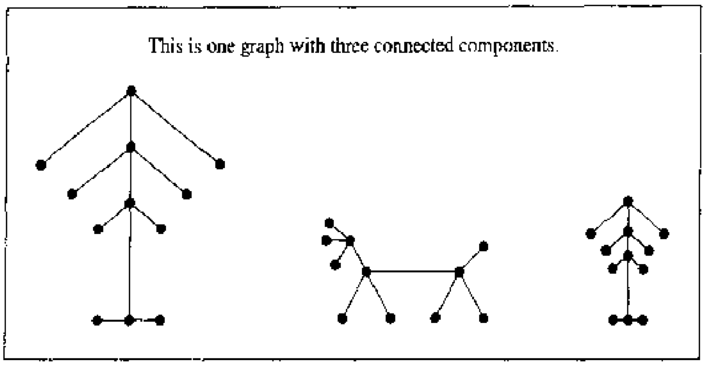


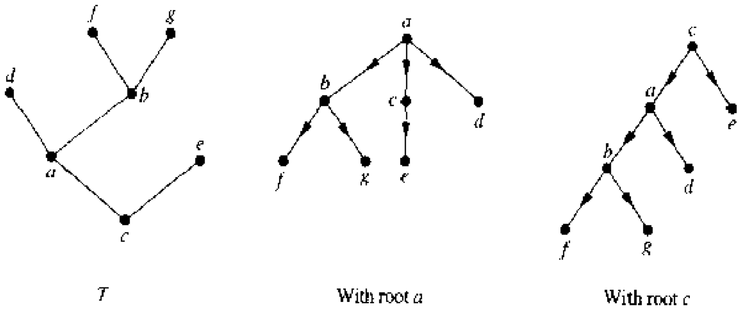
FIGURE 3 **Example of a Forest.**

simple circuit in T. Hence, there is a unique simple path between any two vertices of a tree.

Now assume that there is a unique simple path between any two vertices of a graph T. Then T is connected, since there is a path between any two of its vertices. Furthermore, T can have no simple circuits. To see that this is true, suppose T had a simple circuit that contained the vertices x and y. Then there would be two simple paths between x and y, since the simple circuit is made up of a simple path from x to y and a second simple path from y to x. Hence, a graph with a unique simple path between any two vertices is a tree. ☐

In many applications of trees a particular vertex of a tree is designated as the **root.** Once we specify a root, we can assign a direction to each edge as follows. Since there is a unique path from the root to each vertex of the graph (from Theorem 1), we direct each edge away from the root. Thus, a tree together with its root produces a directed graph called a **rooted tree.** We can change an unrooted tree into a rooted tree by choosing any vertex as the root. Note that different choices of the root produce different rooted trees. For instance, Figure 4 displays the rooted trees formed by designating a to be the root and c to be the root, respectively, in the tree T. We usually draw a rooted tree with its root at the top of the graph. The arrows indicating the directions of the edges in a rooted tree can be omitted, since the choice of root determines the directions of the edges.

The terminology for trees has botanical and genealogical origins. Suppose that T is a rooted tree. If v is a vertex in T other than the root, the **parent** of v is the unique



T With root a With root c

FIGURE 4 **A Tree and Rooted Trees Formed by Designating Two Roots.**

vertex u such that there is a directed edge from u to v (the reader should show that such a vertex is unique). When u is the parent of v, v is called a **child** of u. Vertices with the same parent are called **siblings.** The **ancestors** of a vertex other than the root are the vertices in the path from the root to this vertex, excluding the vertex itself and including the root (that is, its parent, its parent's parent, and so on, until the root is reached). The **descendants** of a vertex v are those vertices that have v as an ancestor. A vertex of a tree is called a **leaf** if it has no children. Vertices that have children are called **internal vertices.** The root is an internal vertex unless it is the only vertex in the graph, in which case it is a leaf.

If a is a vertex in a tree, the **subtree** with a as its root is the subgraph of the tree consisting of a and its descendants and all edges incident to these descendants.

EXAMPLE 2 In the rooted tree T (with root a) shown in Figure 5, find the parent of c, the children of g, the siblings of h, all ancestors of e, all descendants of b, all internal vertices, and all leaves. What is the subtree rooted at g?

Solution: The parent of c is b. The children of g are h, i, and j. The siblings of h are i and j. The ancestors of e are c, b, and a. The descendants of b are c, d, and e. The internal vertices are a, b, c, g, h, and j. The leaves are d, e, f, i, k, l, and m. The subtree rooted at g is shown in Figure 6. ■

Rooted trees with the property that all of their internal vertices have the same number of children are used in many different applications. Later in this chapter we will use such trees to study problems involving searching, sorting, and coding.

DEFINITION 2. **A rooted tree is called an *m-ary tree* if every internal vertex has no more than m children. The tree is called a *full m-ary tree* if every internal vertex has exactly m children. An m-ary tree with $m = 2$ is called a *binary tree.***

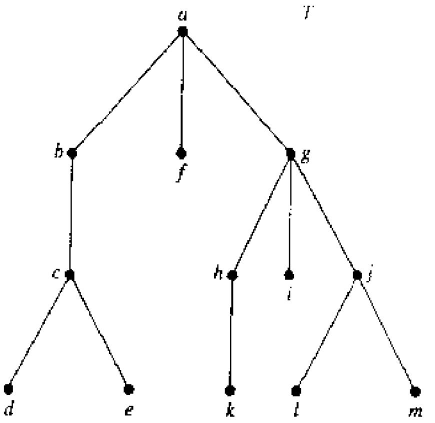


FIGURE 5 A Rooted Tree T.

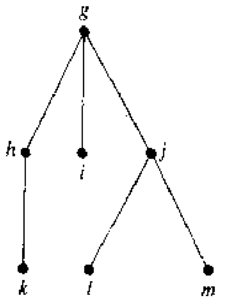


FIGURE 6 The Subtree Rooted At g.

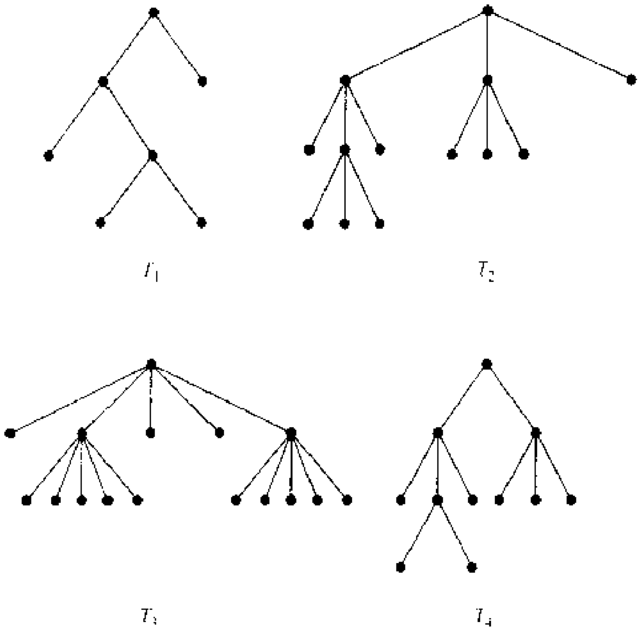


FIGURE 7 Four Rooted Trees.

EXAMPLE 3 Are the rooted trees in Figure 7 full m-ary trees for some positive integer m?

Solution: T_1 is a full binary tree since each of its internal vertices has two children. T_2 is a full 3-ary tree since each of its internal vertices has three children. In T_3 each internal vertex has five children, so T_3 is a full 5-ary tree. T_4 is not a full m-ary tree for any m since some of its internal vertices have two children and others have three children. ■

An **ordered rooted tree** is a rooted tree where the children of each internal vertex are ordered. Ordered rooted trees are drawn so that the children of each internal vertex are shown in order from left to right. Note that a representation of a rooted tree in the conventional way determines an ordering for its edges. We will use such orderings of edges in drawings without explicitly mentioning that we are considering a rooted tree to be ordered.

In an ordered binary tree, if an internal vertex has two children, the first child is called the **left child** and the second child is called the **right child.** The tree rooted at the left child of a vertex is called the **left subtree** of this vertex, and the tree rooted at the right child of a vertex is called the **right subtree** of the vertex. The reader should note that for some applications every vertex of a binary tree, other than the root, is designated as a right or a left child of its parent. This is done even when some vertices have only one child. We will make such designations whenever it is necessary, but not otherwise.

EXAMPLE 4 What are the left and right children of d in the binary tree T shown in Figure 8(a) (where the order is that implied by the drawing)? What are the left and right subtrees of c?

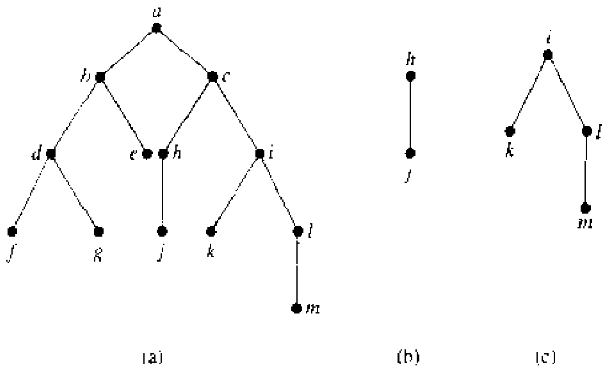


FIGURE 8 A Binary Tree T and Left and Right Subtrees of the Vertex c.

Solution: The left child of d is f and the right child is g. We show the left and right subtrees of c in Figures 8(b) and 8(c), respectively. ■

Just as in the case of graphs, there is no standard terminology used to describe trees, rooted trees, ordered rooted trees, and binary trees. This nonstandard terminology occurs since trees are used extensively throughout computer science, which is a relatively young field. The reader should carefully check meanings given to terms dealing with trees whenever they occur.

TREES AS MODELS

Trees are used as models in such diverse areas as computer science, chemistry, geology, botany, and psychology. We will describe a variety of such models based on trees.

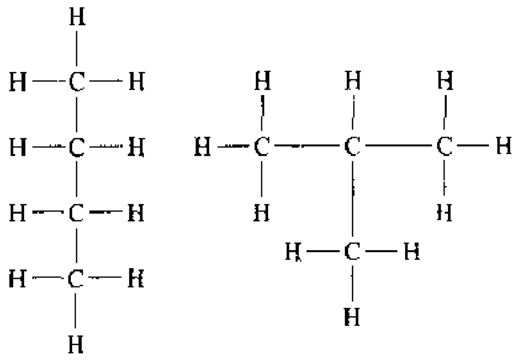
EXAMPLE 5 Saturated Hydrocarbons and Trees Graphs can be used to represent molecules, where atoms are represented by vertices and bonds between them by edges. The English mathematician Arthur Cayley discovered trees in 1857 when he was trying to enumerate the isomers of compounds of the form C_nH_{2n+2}, which are called *saturated hydrocarbons.*

In graph models of saturated hydrocarbons, each carbon atom is represented by a vertex of degree 4, and each hydrogen atom is represented by a vertex of degree 1.

---

web

Arthur Cayley (1821–1895). Arthur Cayley, the son of a merchant, displayed his mathematical talents at an early age with amazing skill in numerical calculations. Cayley entered Trinity College, Cambridge, when he was 17. While in college he developed a passion for reading novels. Cayley excelled at Cambridge and was elected to a 3-year appointment as Fellow of Trinity and assistant tutor. During this time Cayley began his study of n-dimensional geometry and made a variety of contributions to geometry and to analysis. He also developed an interest in mountaineering, which he enjoyed during vacations in Switzerland. Since no position as a mathematician was available to him, Cayley left Cambridge, entering the legal profession and gaining admittance to the bar in 1849. Although Cayley limited his legal work to be able to continue his mathematics research, he developed a reputation as a legal specialist. During his legal career he was able to write more than 300 mathematical papers. In 1863 Cambridge University established a new post in mathematics and offered it to Cayley. He took this job, even though it paid less money than he made as a lawyer.



Butane Isobutane

FIGURE 9 **The Two Isomers of Butane.**

There are $3n+2$ vertices in a graph representing a compound of the form C_nH_{2n+2}. The number of edges in such a graph is half the sum of the degrees of the vertices. Hence, there are $(4n + 2n + 2)/2 = 3n + 1$ edges in this graph. Since the graph is connected and the number of edges is one less than the number of vertices, it must be a tree (see Exercise 9 at the end of this section).

The nonisomorphic trees with n vertices of degree 4 and $2n+2$ of degree 1 represent the different isomers of C_nH_{2n+2}. For instance, when $n = 4$, there are exactly two nonisomorphic trees of this type (the reader should verify this). Hence, there are exactly two different isomers of C_4H_{10}. Their structures are displayed in Figure 9. These two isomers are called butane and isobutane. ■

EXAMPLE 6 Representing Organizations The structure of a large organization can be modeled using a rooted tree. Each vertex in this tree represents a position in the organization. An edge from one vertex to another indicates that the person represented by the initial vertex is the (direct) boss of the person represented by the terminal vertex. The graph shown in Figure 10 displays such a tree. In the organization represented

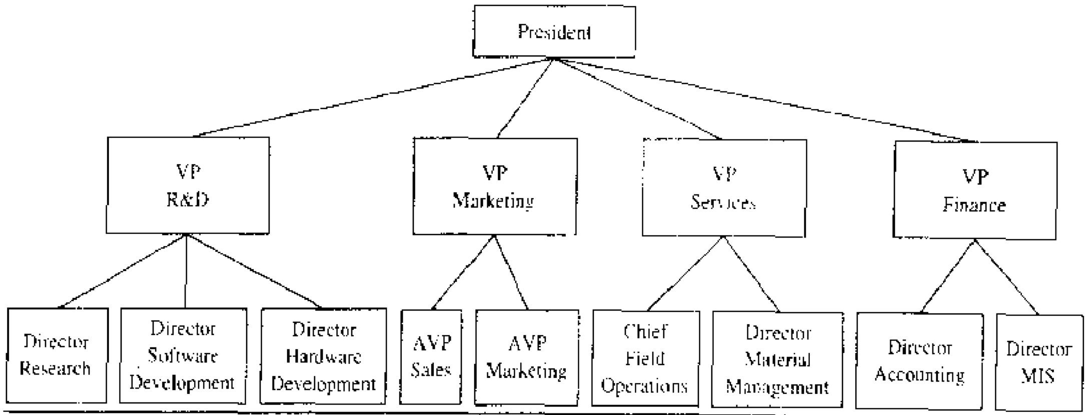

President
VP R&D
VP Marketing
VP Services
VP Finance
Director Research
Director Software Development
Director Hardware Development
AVP Sales
AVP Marketing
Chief Field Operations
Director Material Management
Director Accounting
Director MIS

FIGURE 10 **An Organizational Tree for a Computer Company.**

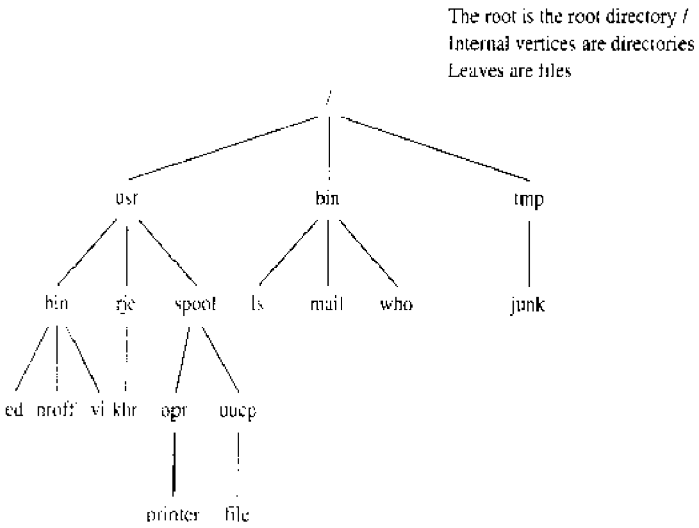


FIGURE 11 **A Computer File System.**

by this tree, the Director of Hardware Development works directly for the Vice President of R&D. ■

EXAMPLE 7 Computer File Systems Files in computer memory can be organized into directories. A directory can contain both files and subdirectories. The root directory contains the entire file system. Thus, a file system may be represented by a rooted tree, where the root represents the root directory, internal vertices represent subdirectories, and leaves represent ordinary files or empty directories. One such file system is shown in Figure 11. In this system, the file khr is in the directory rje. ■

EXAMPLE 8 Tree-Connected Parallel Processors In Example 13 of Section 7.2 we described several interconnection networks for parallel processing. A **tree-connected network** is another important way to interconnect processors. The graph representing such a network is a complete binary tree. Such a network interconnects $n = 2^k - 1$ processors, where k is a positive integer. A processor represented by the vertex v that is not a root or a leaf has 3 two-way connections—one to the processor represented by the parent of v and two to the processors represented by the two children of v. The processor represented by the root has 2 two-way connections to the processors represented by its two children. A processor represented by a leaf v has a single two-way connection to the parent of v. We display a tree-connected network with seven processors in Figure 12.

We will illustrate how a tree-connected network can be used for parallel computation. In particular, we will show how the processors in Figure 12 can be used to add eight numbers, using three steps. In the first step, we add x_1 and x_2 using P_4, x_3 and x_4 using P_5, x_5 and x_6 using P_6, and x_7 and x_8 using P_7. In the second step, we add $x_1 + x_2$ and $x_3 + x_4$ using P_2 and $x_5 + x_6$ and $x_7 + x_8$ using P_3. Finally, in the third step, we add $x_1 + x_2 + x_3 + x_4$ and $x_5 + x_6 + x_7 + x_8$ using P_1. The three steps used to add eight

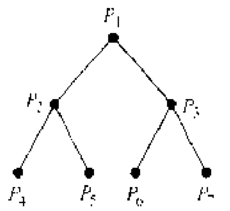


FIGURE 12 A Tree-Connected Network of Seven Processors.

numbers compares favorably to the seven steps required to add eight numbers serially, where the steps are the addition of one number to the sum of the previous numbers in the list. ■

PROPERTIES OF TREES

We will often need results relating the numbers of edges and vertices of various types in trees.

THEOREM 2 **A tree with n vertices has $n - 1$ edges.**

Proof: Choose the vertex r as the root of the tree. We set up a one-to-one correspondence between the edges and the vertices other than r by associating the terminal vertex of an edge to that edge. Since there are $n - 1$ vertices other than r, there are $n - 1$ edges in the tree. □

The number of vertices in a full m-ary tree with a specified number of internal vertices is determined, as the following theorem shows. As in Theorem 2, we will use n to denote the number of vertices in a tree.

THEOREM 3 **A full m-ary tree with i internal vertices contains $n = mi + 1$ vertices.**

Proof: Every vertex, except the root, is the child of an internal vertex. Since each of the i internal vertices has m children, there are mi vertices in the tree other than the root. Therefore, the tree contains $n = mi + 1$ vertices. □

Suppose that T is a full m-ary tree. Let i be the number of internal vertices and l the number of leaves in this tree. Once one of n, i, and l is known, the other two quantities are determined. How to find the other two quantities from the one that is known is given in the following theorem.

THEOREM 4 **A full m-ary tree with**

(i) **n vertices has $i = (n - 1)/m$ internal vertices and $l = [(m - 1)n + 1]/m$ leaves,**
(ii) **i internal vertices has $n = mi + 1$ vertices and $l = (m - 1)i + 1$ leaves,**
(iii) **l leaves has $n = (ml - 1)/(m - 1)$ vertices and $i = (l - 1)/(m - 1)$ internal vertices.**

Proof: Let n represent the number of vertices, i the number of internal vertices, and l the number of leaves. The three parts of the theorem can all be proved using the equality given in Theorem 3, that is, $n = mi + 1$, together with the equality $n = l + i$, which is true because each vertex is either a leaf or an internal vertex. We will prove part (i) here. The proofs of parts (ii) and (iii) are left as exercises for the reader.

Solving for i in $n = mi + 1$ gives $i = (n-1)/m$. Then inserting this expression for i into the equation $n = l + i$ shows that $l = n - i = n - (n-1)/m = [(m-1)n + 1]/m$. □

The following example illustrates how Theorem 4 can be used.

EXAMPLE 9 Suppose that someone starts a chain letter. Each person who receives the letter is asked to send it on to four other people. Some people do this, but others do not send any letters. How many people have seen the letter, including the first person, if no one receives more than one letter and if the chain letter ends after there have been 100 people who read it but did not send it out? How many people sent out the letter?

Solution: The chain letter can be represented using a 4-ary tree. The internal vertices correspond to people who sent out the letter, and the leaves correspond to people who did not send it out. Since 100 people did not send out the letter, the number of leaves in this rooted tree is $l = 100$. Hence, part (iii) of Theorem 4 shows that the number of people who have seen the letter is $n = (4 \cdot 100 - 1)/(4 - 1) = 133$. Also, the number of internal vertices is $133 - 100 = 33$, so that 33 people sent out the letter. ■

It is often desirable to use rooted trees that are "balanced" so that the subtrees at each vertex contain paths of approximately the same length. Some definitions will make this concept clear. The **level** of a vertex v in a rooted tree is the length of the unique path from the root to this vertex. The level of the root is defined to be zero. The **height** of a rooted tree is the maximum of the levels of vertices. In other words, the height of a rooted tree is the length of the longest path from the root to any vertex.

EXAMPLE 10 Find the level of each vertex in the rooted tree shown in Figure 13. What is the height of this tree?

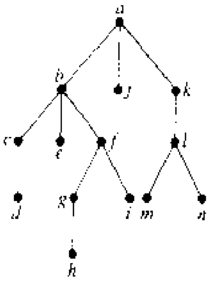


FIGURE 13 A Rooted Tree.

Solution: The root a is at level 0. Vertices b, j, and k are at level 1. Vertices c, e, f, and l are at level 2. Vertices d, g, i, m, and n are at level 3. Finally, vertex h is at level 4. Since the largest level of any vertex is 4, this tree has height 4. ■

A rooted m-ary tree of height h is **balanced** if all leaves are at levels h or $h - 1$.

EXAMPLE 11 Which of the rooted trees shown in Figure 14 are balanced?

Solution: T_1 is balanced, since all its leaves are at levels 3 and 4. However, T_2 is not balanced, since it has leaves at levels 2, 3, and 4. Finally, T_3 is balanced, since all its leaves are at level 3. ■

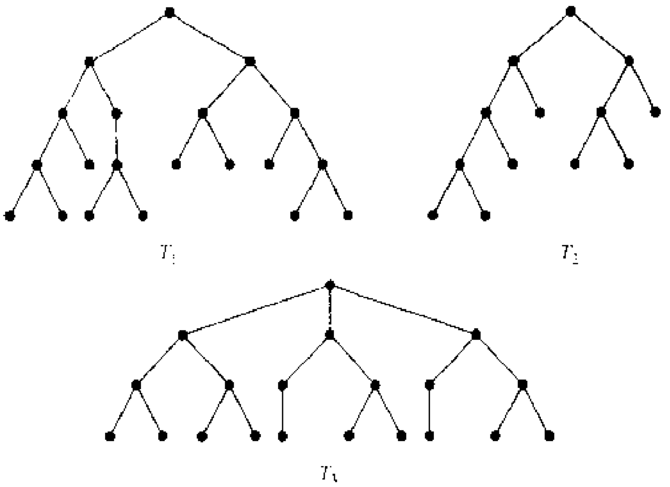


FIGURE 14 **Some Rooted Trees.**

The following results relate the height and the number of leaves in m-ary trees.

THEOREM 5 **There are at most m^h leaves in an m-ary tree of height h.**

Proof: The proof uses mathematical induction on the height. First, consider m-ary trees of height 1. These trees consist of a root with no more than m children, each of which is a leaf. Hence there are no more than $m^1 = m$ leaves in an m-ary tree of height 1. This is the basic step of the inductive argument.

Now assume that the result is true for all m-ary trees of height less than h; this is the inductive hypothesis. Let T be an m-ary tree of height h. The leaves of T are the leaves of the subtrees of T obtained by deleting the edges from the root to each of the vertices at level 1, as shown in Figure 15.

Each of these subtrees has height less than or equal to $h - 1$. So by the inductive hypothesis, each of these rooted trees has at most m^{h-1} leaves. Since there are at most m such subtrees, each with a maximum of m^{h-1} leaves, there are at most $m \cdot m^{h-1} = m^h$ leaves in the rooted tree. This finishes the inductive argument. □

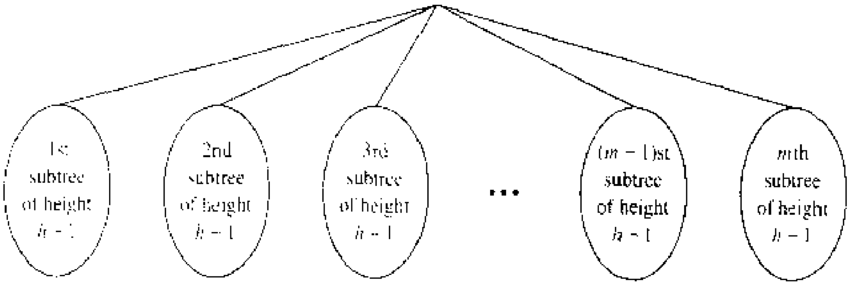

1st subtree of height $h-1$
2nd subtree of height $h-1$
3rd subtree of height $h-1$
...
$(m-1)$st subtree of height $h-1$
mth subtree of height $h-1$

FIGURE 15 **The Inductive Step of the Proof.**

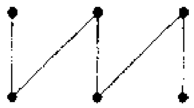
COROLLARY 1 **If an m-ary tree of height h has l leaves, then $h \geq \lceil \log_m l \rceil$. If the m-ary tree is full and balanced, then $h = \lceil \log_m l \rceil$. (We are using the ceiling function here. Recall that $\lceil x \rceil$ is the smallest integer greater than or equal to x.)**

Proof: We know that $l \leq m^h$ from Theorem 5. Taking logarithms to the base m shows that $\log_m l \leq h$. Since h is an integer, we have $h \geq \lceil \log_m l \rceil$. Now suppose that the tree is balanced. Then each leaf is at level h or $h - 1$, and since the height is h, there is at least one leaf at level h. It follows that there must be more than m^{h-1} leaves (see Exercise 24 at the end of this section). Since $l \leq m^h$, we have $m^{h-1} < l \leq m^h$. Taking logarithms to the base m in this inequality gives $h - 1 < \log_m l \leq h$. Hence, $h = \lceil \log_m l \rceil$. □

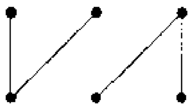
Exercises

1. Which of the following graphs are trees?

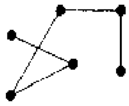
a)



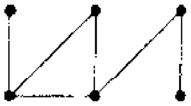
b)



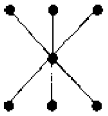
c)



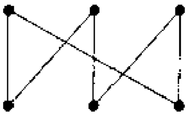
d)



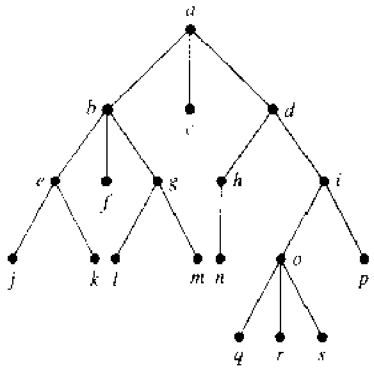
e)



f)



2. Answer the following questions about the rooted tree illustrated.



a) Which vertex is the root?
b) Which vertices are internal?
c) Which vertices are leaves?
d) Which vertices are children of j?
e) Which vertex is the parent of h?
f) Which vertices are siblings of o?
g) Which vertices are ancestors of m?
h) Which vertices are descendants of b?

3. Is the rooted tree in Exercise 2 a full m-ary tree for some positive integer m?

4. What is the level of each vertex of the tree in Exercise 2?

5. Draw the subtree of the tree in Exercise 2 that is rooted at
a) a.
b) c.
c) e.

***6.** How many nonisomorphic unrooted trees are there with n vertices if
a) $n = 3$?
b) $n = 4$?
c) $n = 5$?

***7.** Answer the same question as that given in Exercise 6 for rooted trees (using isomorphism for directed graphs).

***8.** Show that a simple graph is a tree if and only if it is connected, but the deletion of any of its edges produces a graph that is not connected.

***9.** Let G be a simple graph with n vertices. Show that G is a tree if and only if G is connected and has $n - 1$ edges.

10. Which complete bipartite graphs $K_{m,n}$, where m and n are positive integers, are trees?

11. How many edges does a tree with 10,000 vertices have?

12. How many vertices does a full 5-ary tree with 100 internal vertices have?

13. How many edges does a full binary tree with 1000 internal vertices have?

14. How many leaves does a full 3-ary tree with 100 vertices have?

15. Suppose 1000 people enter a chess tournament. Use a rooted tree model of the tournament to determine how many games must be played to determine a champion, if a player is eliminated after one loss and games are played until only one entrant has not lost. (Assume there are no ties.)

16. A chain letter starts when a person sends a letter to five others. Each person who receives the letter either sends it to five other people who have never received it or does not send it to anyone. Suppose that 10,000 people send out the letter before the chain ends and that no one receives more than one letter. How many people receive the letter, and how many do not send it out?

17. A chain letter starts with a person sending a letter out to 10 others. Each person is asked to send the letter out to 10 others, and each letter contains a list of the previous 6 people in the chain. Unless there are fewer than six names in the list, each person sends one dollar to the first person in this list, removes the name of this person from the list, moves up each of the other five names one position, and inserts his or her name at the end of this list. If no person breaks the chain and no one receives more than one letter, how much money will a person in the chain ultimately receive?

***18.** Either draw a full m-ary tree with 76 leaves and height 3 where m is a positive integer or show that no such tree exists.

***19.** Either draw a full m-ary tree with 84 leaves and height 3 where m is a positive integer or show that no such tree exists.

***20.** A full m-ary tree T has 81 leaves and height 4.

a) Give the upper and lower bounds for m.
b) What is m if T is also balanced?

A **complete m-ary tree** is a full m-ary tree where every leaf is at the same level.

21. Construct a complete binary tree of height 4 and a complete 3-ary tree of height 3.

22. How many vertices and how many leaves does a complete m-ary tree of height h have?

23. Prove

a) part (ii) of Theorem 4.
b) part (iii) of Theorem 4.

☞ **24.** Show that a full m-ary balanced tree of height h has more than m^{h-1} leaves.

25. How many edges are there in a forest of t trees containing a total of n vertices?

26. Explain how a tree can be used to represent the table of contents of a book organized into chapters, where each chapter is organized into sections, and each section is organized into subsections.

27. How many different isomers do the following saturated hydrocarbons have?

a) C_3H_8 **b)** C_5H_{12} **c)** C_6H_{14}

28. What does each of the following represent in an organizational tree?

a) the parent of a vertex
b) a child of a vertex
c) a sibling of a vertex
d) the ancestors of a vertex
e) the descendants of a vertex
f) the level of a vertex
g) the height of the tree

29. Answer the same questions as those given in Exercise 28 for a rooted tree representing a computer file system.

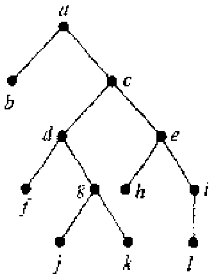
30. **a)** Draw the complete binary tree with 15 vertices that represents a tree-connected network of 15 processors.
b) Show how 16 numbers can be added using the 15 processors in part (a) using four steps.

31. Let n be a power of 2. Show that n numbers can be added in $\log n$ steps using a tree-connected network of $n - 1$ processors.

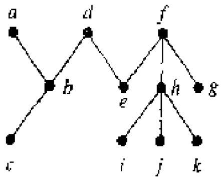
***32.** A **labeled tree** is a tree where each vertex is assigned a label. Two labeled trees are considered isomorphic when there is an isomorphism between them that preserves the labels of vertices. How many nonisomorphic trees are there with three vertices labeled with different integers from the set $\{1, 2, 3\}$? How many nonisomorphic trees are there with four vertices labeled with different integers from the set $\{1, 2, 3, 4\}$?

The **eccentricity** of a vertex in an unrooted tree is the length of the longest simple path beginning at this vertex. A vertex is called a **center** if no vertex in the tree has smaller eccentricity than this vertex. In Exercises 33–35 find every vertex that is a center in the given tree.

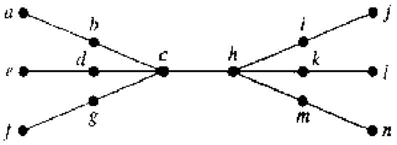
33.



34.



35.



36. Show that a center should be chosen as the root to produce a rooted tree of minimal height from an unrooted tree.

*37. Show that a tree has either one center or two centers that are adjacent.

38. Show that every tree can be colored using two colors.

The **rooted Fibonacci trees** T_n are defined recursively in the following way. T_1 and T_2 are both the rooted tree consisting of a single vertex, and for $n = 3, 4, \ldots$, the rooted tree T_n is constructed from a root with T_{n-1} as its left subtree and T_{n-2} as its right subtree.

39. Draw the first seven rooted Fibonacci trees.

*40. How many vertices, leaves, and internal vertices does the rooted Fibonacci tree T_n have, where n is a positive integer? What is its height?

8.2

Applications of Trees

INTRODUCTION

We will discuss three problems that can be studied using trees. The first problem is: How should items in a list be stored so that an item can be easily located? The second problem is: What series of decisions should be made to find an object with a certain property in a collection of objects of a certain type? The third problem is: How should a set of characters be efficiently coded by bit strings?

BINARY SEARCH TREES

web

Searching for items in a list is one of the most important tasks that arises in computer science. Our primary goal is to implement a searching algorithm that finds items efficiently when the items are totally ordered. This can be accomplished through the use of a **binary search tree,** which is a binary tree in which each child of a vertex is designated as a right or left child, no vertex has more than one right child or left child, and each vertex is labeled with a key, which is one of the items. Furthermore, vertices are assigned keys so that the key of a vertex is both larger than the keys of all vertices in its left subtree and smaller than the keys of all vertices in its right subtree.

The following recursive procedure is used to form the binary search tree for a list of items. Start with a tree containing just one vertex, namely, the root. The first item in the list is assigned as the key of the root. To add a new item, first compare it with the keys of vertices already in the tree, starting at the root and moving to the left if the item is less than the key of the respective vertex if this vertex has a left child, or moving to the right if the item is greater than the key of the respective vertex if this vertex has a right child. When the item is less than the respective vertex and this vertex has no left child, then a new vertex with this item as its key is inserted as a new left child. Similarly, when the item is greater than the respective vertex and this vertex has no right child, then a new vertex with this item as its key is inserted as a new right child. We illustrate this procedure with the following example.

EXAMPLE 1 Form a binary search tree for the words *mathematics, physics, geography, zoology, meteorology, geology, psychology,* and *chemistry* (using alphabetical order).

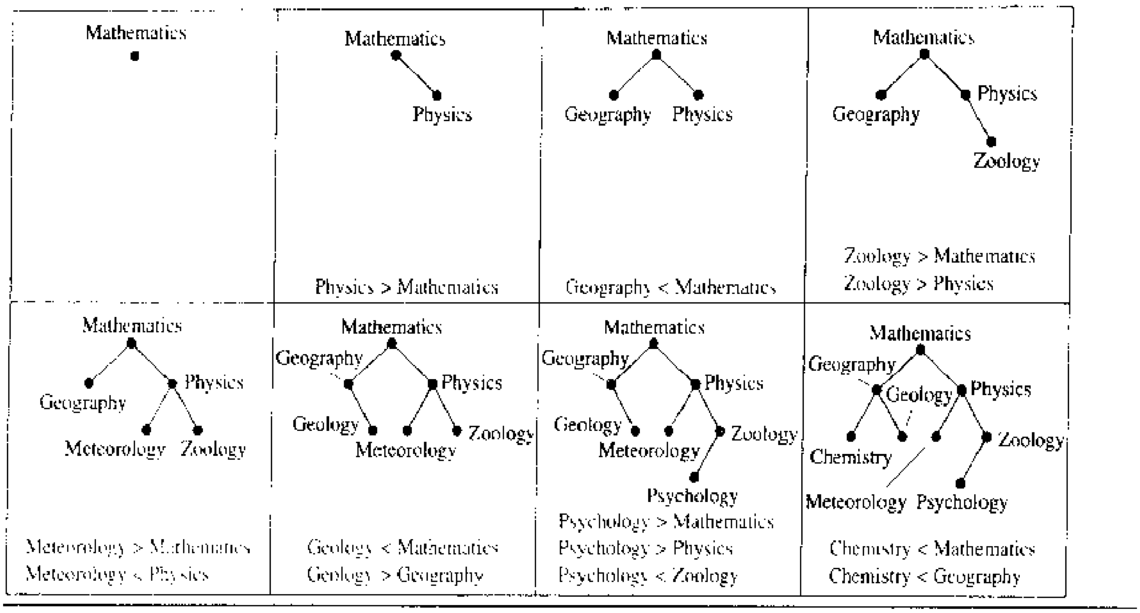

Mathematics
Mathematics
Physics
Physics > Mathematics
Mathematics
Geography
Physics
Geography < Mathematics
Mathematics
Geography
Physics
Zoology
Zoology > Mathematics
Zoology > Physics
Mathematics
Geography
Physics
Meteorology
Zoology
Meteorology > Mathematics
Meteorology < Physics
Mathematics
Geography
Physics
Geology
Meteorology
Zoology
Geology < Mathematics
Geology > Geography
Mathematics
Geography
Physics
Geology
Meteorology
Zoology
Psychology
Psychology > Mathematics
Psychology > Physics
Psychology < Zoology
Mathematics
Geography
Geology
Physics
Chemistry
Zoology
Meteorology
Psychology
Chemistry < Mathematics
Chemistry < Geography

FIGURE 1 **Constructing a Binary Search Tree.**

Solution: Figure 1 displays the steps used to construct this binary search tree. The word *mathematics* is the key of the root. Since *physics* comes after *mathematics* (in alphabetical order), add a right child of the root with key *physics*. Since *geography* comes before *mathematics*, add a left child of the root with key *geography*. Next, add a right child of the vertex with key *physics*, and assign it the key *zoology*, since *zoology* comes after *mathematics* and after *physics*. Similarly, add a left child of the vertex with key *physics* and assign this new vertex the key *meteorology*. Add a right child of the vertex with key *geography* and assign this new vertex the key *geology*. Add a left child of the vertex with key *zoology* and assign it the key *psychology*. Add a left child of the vertex with key *geography* and assign it the key *chemistry*. (The reader should work through all the comparisons needed at each step.) ■

To locate an item we try to add it to a binary search tree. We locate it if it is present. Algorithm 1 gives pseudocode for locating an item in a binary search tree and adding a new vertex with this item as its key if the item is not found. Algorithm 1 locates x if it is already the key of a vertex. When x is not a key, a new vertex with key x is added to the tree. In the pseudocode, v is the vertex that has x as its key, and *label*(v) represents the key of vertex v.

We will now determine the computational complexity of this procedure. Suppose we have a binary search tree T for a list of n items. We can form a full binary tree U from T by adding unlabeled vertices whenever necessary so that every vertex with a key has two children. This is illustrated in Figure 2. Once we have done this, we can easily locate or add a new item as a key without adding a vertex.

The most comparisons needed to add a new item is the length of the longest path in U from the root to a leaf. The internal vertices of U are the vertices of T. It follows that U has n internal vertices. We can now use part (ii) of Theorem 4 in Section 8.1 to conclude that U has $n + 1$ leaves. Using Corollary 1 of Section 8.1, we see that the

ALGORITHM 1 **Binary Search Tree Algorithm.**

```
procedure insertion($T$: binary search tree, $x$: item)
$v := $ root of $T$
{a vertex not present in $T$ has the value null}
while $v \neq$ null and label($v$) $\neq x$
begin
  if $x <$ label($v$) then
    if left child of $v \neq$ null then $v :=$ left child of $v$
    else add new vertex as a left child of $v$ and set $v :=$ null
  else
    if right child of $v \neq$ null then $v :=$ right child of $v$
    else add new vertex as a right child of $v$ to $T$ and set $v :=$ null
end
if root of $T$ = null then add a vertex $r$ to the tree and label it with $x$
else if label($v$) $\neq x$ then label new vertex with $x$
{$v$ = location of $x$}
```

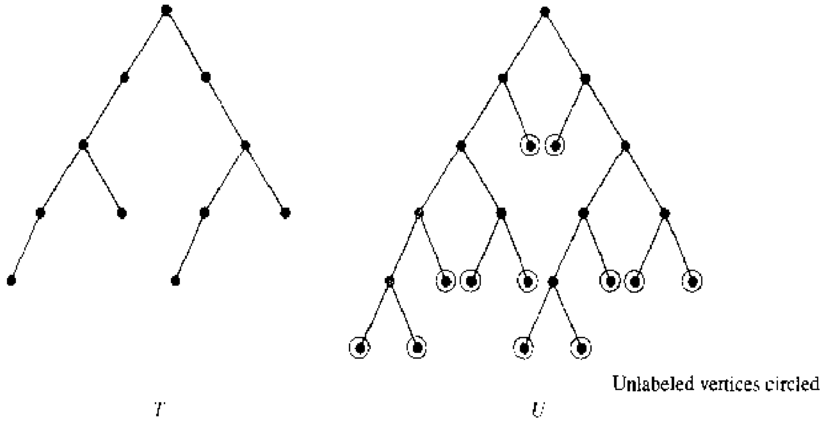


FIGURE 2 Adding Unlabeled Vertices to Make a Binary Search Tree Full.

height of U is greater than or equal to $h = \lceil \log(n+1) \rceil$. Consequently, it is necessary to perform at least $\lceil \log(n+1) \rceil$ comparisons to add some item. Note that if U is balanced, its height is $\lceil \log(n+1) \rceil$ (from Corollary 1 of Section 8.1). Thus, if a binary search tree is balanced, locating or adding an item requires no more than $\lceil \log(n+1) \rceil$ comparisons. A binary search tree can become unbalanced as items are added to it. Since balanced binary search trees give optimal worst-case complexity for binary searching, algorithms have been devised that rebalance binary search trees as items are added. The interested reader can consult references on data structures for the description of such algorithms.

DECISION TREES

Rooted trees can be used to model problems in which a series of decisions leads to a solution. For instance, a binary search tree can be used to locate items based on a series of comparisons, where each comparison tells us whether we have located the item, or whether we should go right or left in a subtree. A rooted tree in which each internal vertex corresponds to a decision, with a subtree at these vertices for each possible outcome of the decision, is called a **decision tree.** The possible solutions of the problem correspond to the paths to the leaves of this rooted tree. The next example illustrates an application of decision trees.

EXAMPLE 2 Suppose there are seven coins, all with the same weight, and a counterfeit coin that weighs less than the others. How many weighings are necessary using a balance scale to determine which of the eight coins is the counterfeit one? Give an algorithm for finding this counterfeit coin.

Solution: There are three possibilities for each weighing on a balance scale. The two pans can have equal weight, the first pan can be heavier, or the second pan can be heavier. Consequently, the decision tree for the sequence of weighings is a 3-ary tree. There are at least eight leaves in the decision tree since there are eight possible outcomes (since each of the eight coins can be the counterfeit lighter coin), and each possible outcome must be represented by at least one leaf. The largest number of weighings needed to determine the counterfeit coin is the height of the decision tree. From Corollary 1 of Section 8.1 it follows that the height of the decision tree is at least $\lceil \log_3 8 \rceil = 2$. Hence, at least two weighings are needed.

It is possible to determine the counterfeit coin using two weighings. The decision tree that illustrates how this is done is shown in Figure 3. ■

In Section 4 of this chapter we will study sorting algorithms using decision trees.

PREFIX CODES

Consider the problem of using bit strings to encode the letters of the English alphabet (where no distinction is made between lowercase and uppercase letters). We can represent each letter with a bit string of length five, since there are only 26 letters and there are 32 bit strings of length five. The total number of bits used to encode data is five

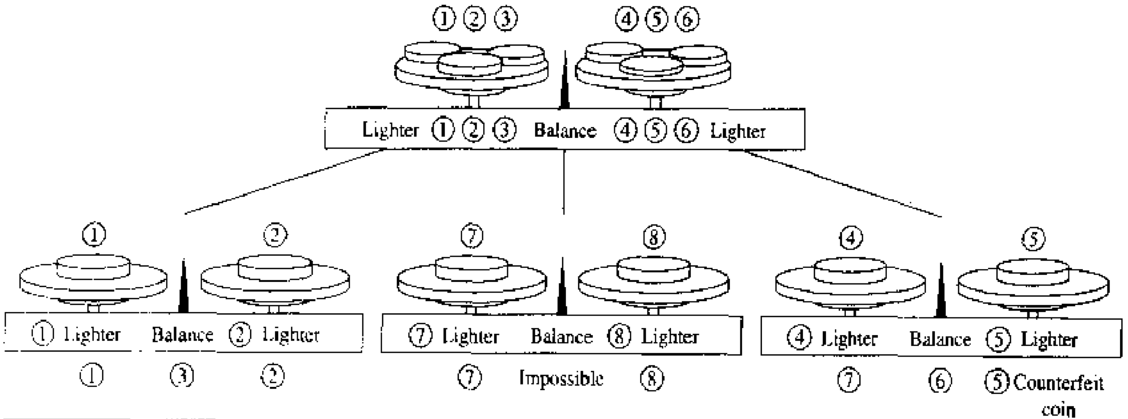

①②③
④⑤⑥
Lighter ①②③ Balance ④⑤⑥ Lighter
①
②
⑦
⑧
④
⑤
① Lighter Balance ② Lighter
⑦ Lighter Balance ⑧ Lighter
④ Lighter Balance ⑤ Lighter
① ③ ②
⑦ Impossible ⑧
⑦ ⑥ ⑤ Counterfeit coin

FIGURE 3 A Decision Tree for Locating a Counterfeit Coin.

times the number of characters in the text when each character is encoded with five bits. Is it possible to find a coding scheme of these letters so that, when data are coded, fewer bits are used? We can save memory and reduce transmittal time if this can be done.

Consider using bit strings of different lengths to encode letters. Letters that occur more frequently should be encoded using short bit strings, and longer bit strings should be used to encode rarely occurring letters. When letters are encoded using varying numbers of bits, some method must be used to determine where the bits for each character start and end. For instance, if *e* were encoded with 0, *a* with 1, and *t* with 01, then the bit string 0101 could correspond to *eat*, *tea*, *eaea*, or *tt*.

To ensure that no bit string corresponds to more than one sequence of letters, the bit string for a letter must never occur as the first part of the bit string for another letter. Codes with this property are called **prefix codes.** For instance, the encoding of *e* as 0, *a* as 10, and *t* as 11 is a prefix code. A word can be recovered from the unique bit string that encodes its letters. For example, the string 10110 is the encoding of *ate*. To see this, note that the initial 1 does not represent a character, but 10 does represent *a* (and could not be the first part of the bit string of another letter). Then, the next 1 does not represent a character, but 11 does represent *t*. The final bit, 0, represents *e*.

A prefix code can be represented using a binary tree, where the characters are the labels of the leaves in the tree. The edges of the tree are labeled so that an edge leading to a left child is assigned a 0 and an edge leading to a right child is assigned a 1. The bit string used to encode a character is the sequence of labels of the edges in the unique path from the root to the leaf that has this character as its label. For instance, the tree in Figure 4 represents the encoding of *e* by 0, *a* by 10, *t* by 110, *n* by 1110, and *s* by 1111.

The tree representing a code can be used to decode a bit string. For instance, consider the word encoded by 11111011100 using the code in Figure 4. This bit string can be decoded by starting at the root, using the sequence of bits to form a path that stops when a leaf is reached. Each 0 bit takes the path down the edge leading to the left child of the last vertex in the path, and each 1 bit corresponds to the right child of this vertex. Consequently, the initial 1111 corresponds to the path starting at the root, going right four times, leading to a leaf in the graph that has *s* as its label, since the string 1111 is the code for *s*. Continuing with the fifth bit, we reach a leaf next after going right then left, when the vertex labeled with *a*, which is encoded by 10, is visited. Starting with the seventh bit, we reach a leaf next after going right three times and then left, when the vertex labeled with *n*, which is encoded by 1110, is visited. Finally, the last bit, 0, leads to the leaf that is labeled with *e*. Therefore, the original word is *sane*.

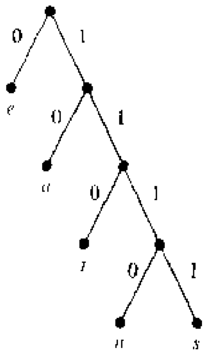


FIGURE 4 **The Binary Tree with a Prefix Code.**

We can construct a prefix code from any binary tree where the left edge at each internal vertex is labeled by 0 and the right edge by a 1 and where the leaves are labeled by characters. Characters are encoded with the bit string constructed using the labels of the edges in the unique path from the root to the leaves.

There are algorithms, such as Huffman coding, that can be used to produce efficient codes based on the frequencies of occurrences of characters. We will not present the details of such algorithms here. (The interested reader can find the details of such algorithms in the references given for this section at the end of the book.)

Exercises

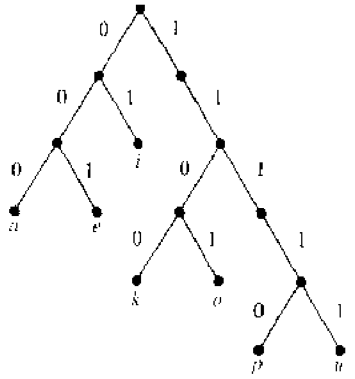
1. Build a binary search tree for the words *banana, peach, apple, pear, coconut, mango,* and *papaya* using alphabetical order.
2. Build a binary search tree for the words *oenology, phrenology, campanology, ornithology, ichthyology, limnology, alchemy,* and *astrology* using alphabetical order.
3. How many comparisons are needed to locate or to add each of the following words in the search tree for Exercise 1, starting fresh each time?
   a) *pear* b) *banana*
   c) *kumquat* d) *orange*
4. How many comparisons are needed to locate or to add each of the following words in the search tree for Exercise 2, starting fresh each time?
   a) *palmistry*
   b) *etymology*
   c) *paleontology*
   d) *glaciology*
5. Using alphabetical order, construct a binary search tree for the words in the sentence, "*The quick brown fox jumps over the lazy dog.*"
6. How many weighings of a balance scale are needed to find a lighter counterfeit coin among four coins? Describe an algorithm to find the lighter coin using this number of weighings.
7. How many weighings of a balance scale are needed to find a counterfeit coin among four coins if the counterfeit coin may be either heavier or lighter than the others? Describe an algorithm to find the counterfeit coin using this number of weighings.

*8. How many weighings of a balance scale are needed to find a counterfeit coin among eight coins if the counterfeit coin is either heavier or lighter than the others? Describe an algorithm to find the counterfeit coin using this number of weighings.

*9. How many weighings of a balance scale are needed to find a counterfeit coin among 12 coins if the counterfeit coin is lighter than the others? Describe an algorithm to find the lighter coin using this number of weighings.

*10. One of four coins may be counterfeit. If it is counterfeit, it may be lighter or heavier than the others. How many weighings are needed, using a balance scale, to determine whether there is a counterfeit coin, and if there is, whether it is lighter or heavier than the others? Describe an algorithm to find the counterfeit coin and determine whether it is lighter or heavier using this number of weighings.

11. Which of the following codes are prefix codes?
    a) *a*:11, *e*:00, *t*:10, *s*:01
    b) *a*:0, *e*:1, *t*:01, *s*:001
    c) *a*:101, *e*:11, *t*:001, *s*:011, *n*:010
    d) *a*:010, *e*:11, *t*:011, *s*:1011, *n*:1001, *i*:10101
12. Construct the binary tree with prefix codes representing the following coding schemes.
    a) *a*:11, *e*:0, *t*:101, *s*:100
    b) *a*:1, *e*:01, *t*:001, *s*:0001, *n*:00001
    c) *a*:1010, *e*:0, *t*:11, *s*:1011, *n*:1001, *i*:10001
13. What are the codes for *a, e, i, k, o, p,* and *u* if the coding scheme is represented by the following tree?



14. Given the coding scheme *a*:001, *b*:0001, *e*:1, *r*:0000, *s*:0100, *t*:011, *x*:01010, find the word represented by
    a) 01110100011.
    b) 0001110000.
    c) 0100101010.
    d) 01100101010.

8.3

Tree Traversal

INTRODUCTION

web

Ordered rooted trees are often used to store information. We need procedures for visiting each vertex of an ordered rooted tree to access data. We will describe several important algorithms for visiting all the vertices of an ordered rooted tree. Ordered rooted trees can also be used to represent various types of expressions, such as arithmetic expressions involving numbers, variables, and operations. The different listings of the vertices of ordered rooted trees used to represent expressions are useful in the evaluation of these expressions.

UNIVERSAL ADDRESS SYSTEMS

Procedures for traversing all vertices of an ordered rooted tree rely on the orderings of children. In ordered rooted trees, the children of an internal vertex are shown from left to right in the drawings representing these directed graphs.

We will describe one way to order totally the vertices of an ordered rooted tree. To produce this ordering, we must first label all the vertices. We do this recursively as follows.

1. Label the root with the integer 0. Then label its k children (at level 1) from left to right with $1, 2, 3, \ldots, k$.

2. For each vertex v at level n with label A, label its k_v children, as they are drawn from left to right, with $A.1, A.2, \ldots, A.k_v$.

Following this procedure, a vertex v at level n, for $n \geq 1$, is labeled $x_1.x_2.\cdots.x_n$, where the unique path from the root to v goes through the x_1th vertex at level 1, the x_2th vertex at level 2, and so on. This labeling is called the **universal address system** of the ordered rooted tree.

We can totally order the vertices using the lexicographic ordering of their labels in the universal address system. The vertex labeled $x_1.x_2.\cdots.x_n$ is less than the vertex labeled $y_1.y_2.\cdots.y_m$ if there is an i, $0 \leq i \leq n$, with $x_1 = y_1, x_2 = y_2, \ldots, x_{i-1} = y_{i-1}$, and $x_i < y_i$; or if $n < m$ and $x_i = y_i$ for $i = 1, 2, \ldots, n$.

EXAMPLE 1 We display the labelings of the universal address system next to the vertices in the ordered rooted tree shown in Figure 1. The lexicographic ordering of the labelings is

$$0 < 1 < 1.1 < 1.2 < 1.3 < 2 < 3 < 3.1 < 3.1.1 < 3.1.2 < 3.1.2.1 < 3.1.2.2 < 3.1.2.3 < 3.1.2.4 < 3.1.3 < 3.2 < 4 < 4.1 < 5 < 5.1 < 5.1.1 < 5.2 < 5.3$$ ■

TRAVERSAL ALGORITHMS

Procedures for systematically visiting every vertex of an ordered rooted tree are called **traversal algorithms.** We will describe three of the most commonly used such algorithms, **preorder traversal, inorder traversal,** and **postorder traversal.** Each of these algorithms can be defined recursively. We first define preorder traversal.

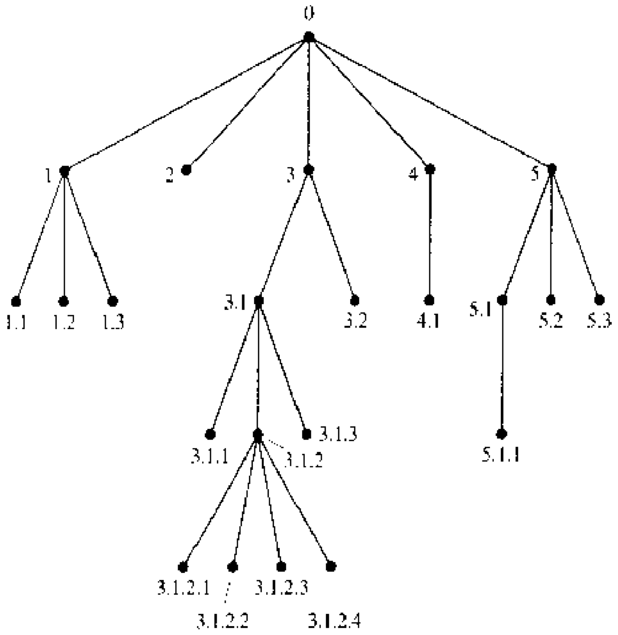

0
1
2
3
4
5
1.1
1.2
1.3
3.1
3.2
4.1
5.1
5.2
5.3
3.1.1
3.1.2
3.1.3
5.1.1
3.1.2.1
3.1.2.2
3.1.2.3
3.1.2.4

FIGURE 1 **The Universal Address System of an Ordered Rooted Tree.**

DEFINITION 1. Let T be an ordered rooted tree with root r. If T consists only of r, then r is the *preorder traversal* of T. Otherwise, suppose that $T_1, T_2, \ldots, T_n$ are the subtrees at r from left to right in T. The *preorder traversal* begins by visiting r. It continues by traversing T_1 in preorder, then T_2 in preorder, and so on, until T_n is traversed in preorder.

The reader should verify that the preorder traversal of an ordered rooted tree gives the same ordering of the vertices as the ordering obtained using a universal address system. Figure 2 indicates how a preorder traversal is carried out.

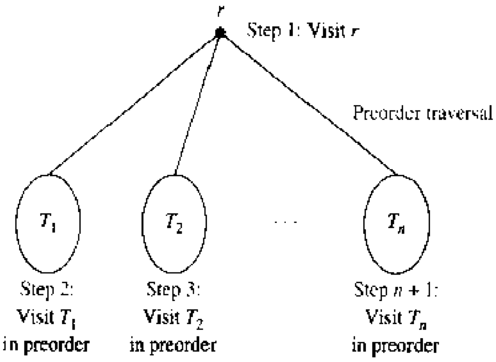

r
Step 1: Visit r
Preorder traversal
T1
T2
. . .
Tn
Step 2: Visit T1 in preorder
Step 3: Visit T2 in preorder
Step n + 1: Visit Tn in preorder

FIGURE 2 **Preorder Traversal.**

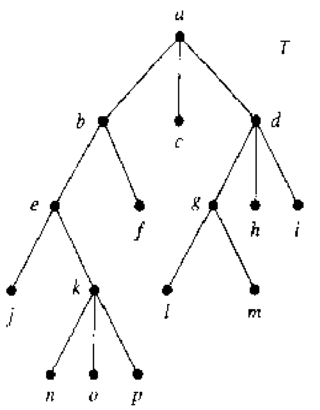


FIGURE 3 The Ordered Rooted Tree T.

The following example illustrates preorder traversal.

EXAMPLE 2 In which order does a preorder traversal visit the vertices in the ordered rooted tree T shown in Figure 3?

Solution: The steps of the preorder traversal of T are shown in Figure 4. We traverse T in preorder by first listing the root a, followed by the preorder list of the subtree with root b, the preorder list of the subtree with root c (which is just c) and the preorder list of the subtree with root d.

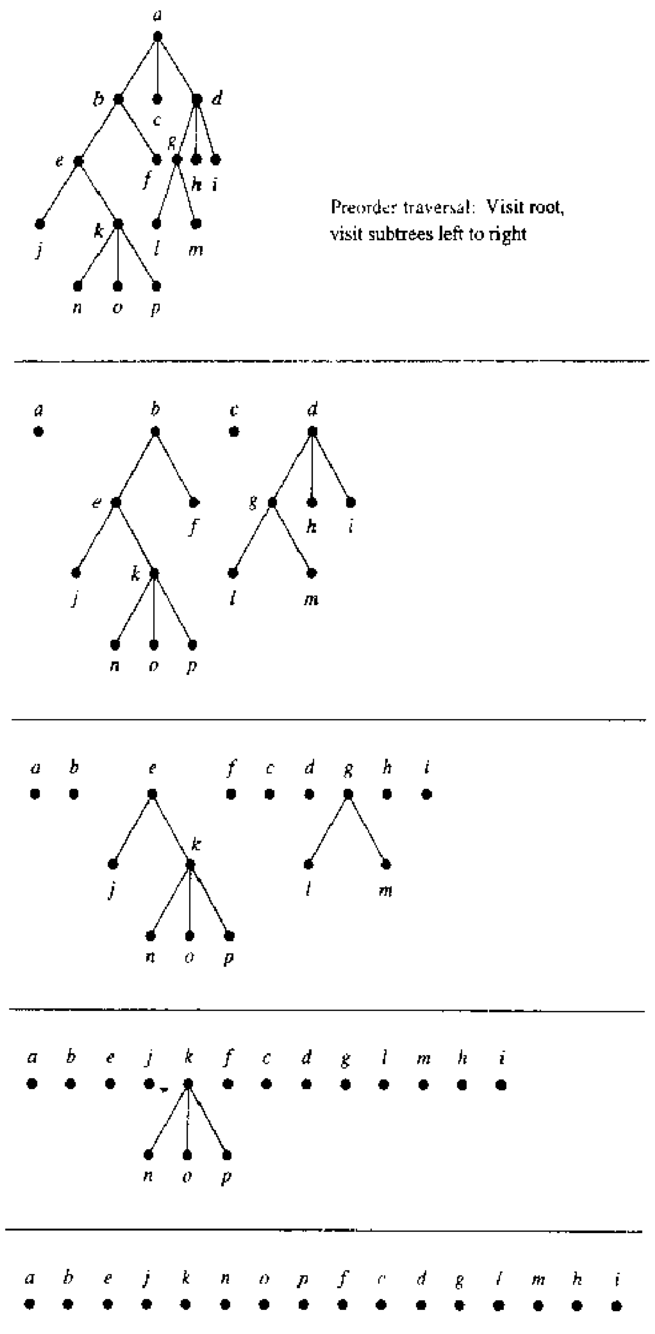
The preorder list of the subtree with root b begins by listing b, then the vertices of the subtree with root e in preorder, and then the subtree with root f in preorder (which is just f). The preorder list of the subtree with root d begins by listing d, followed by the preorder list of the subtree with root g, followed by the subtree with root h (which is just h), followed by the subtree with root i (which is just i).

The preorder list of the subtree with root e begins by listing e, followed by the preorder listing of the subtree with root j (which is just j), followed by the preorder listing of the subtree with root k. The preorder listing of the subtree with root g is g followed by l, followed by m. The preorder listing of the subtree with root k is k, n, o, p. Consequently, the preorder traversal of T is a, b, e, j, k, n, o, p, f, c, d, g, l, m, h, i. ■

We will now define inorder traversal.

DEFINITION 2. Let T be an ordered rooted tree with root r. If T consists only of r, then r is the *inorder traversal* of T. Otherwise, suppose that $T_1, T_2, \ldots, T_n$ are the subtrees at r from left to right. The *inorder traversal* begins by traversing T_1 in inorder, then visiting r. It continues by traversing T_2 in inorder, then T_3 in inorder, $\ldots$, and finally T_n in inorder.

Figure 5 indicates how inorder traversal is carried out.


a b c d e f g h i j k l m n o p
a b c d e f g h i j k l m n o p
a b e f c d g h i j k l m n o p
a b e j k f c d g l m h i n o p
a b e j k n o p f c d g l m h i

Preorder traversal: Visit root, visit subtrees left to right

FIGURE 4 The Preorder Traversal of *T*.

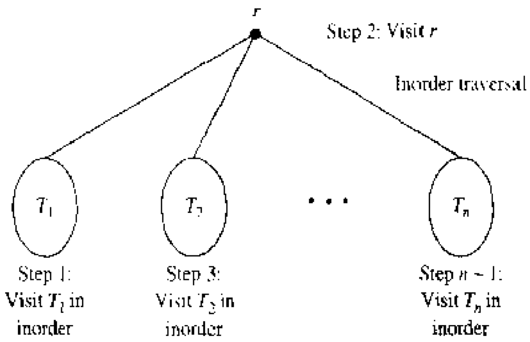

r
Step 2: Visit r
Inorder traversal
T_1
T_2
T_n
Step 1: Visit T_1 in inorder
Step 3: Visit T_2 in inorder
Step n + 1: Visit T_n in inorder

FIGURE 5 **Inorder Traversal.**

The following example illustrates how inorder traversal is carried out.

EXAMPLE 3 In which order does an inorder traversal visit the vertices of the ordered rooted tree T in Figure 3?

Solution: The steps of the inorder traversal of the ordered rooted tree T are shown in Figure 6. The inorder traversal begins with an inorder traversal of the subtree with root b, the root a, the inorder listing of the subtree with root c, which is just c, and the inorder listing of the subtree with root d.

The inorder listing of the subtree with root b begins with the inorder listing of the subtree with root e, the root b, and f. The inorder listing of the subtree with root d begins with the inorder listing of the subtree with root g, followed by the root d, followed by h, followed by i.

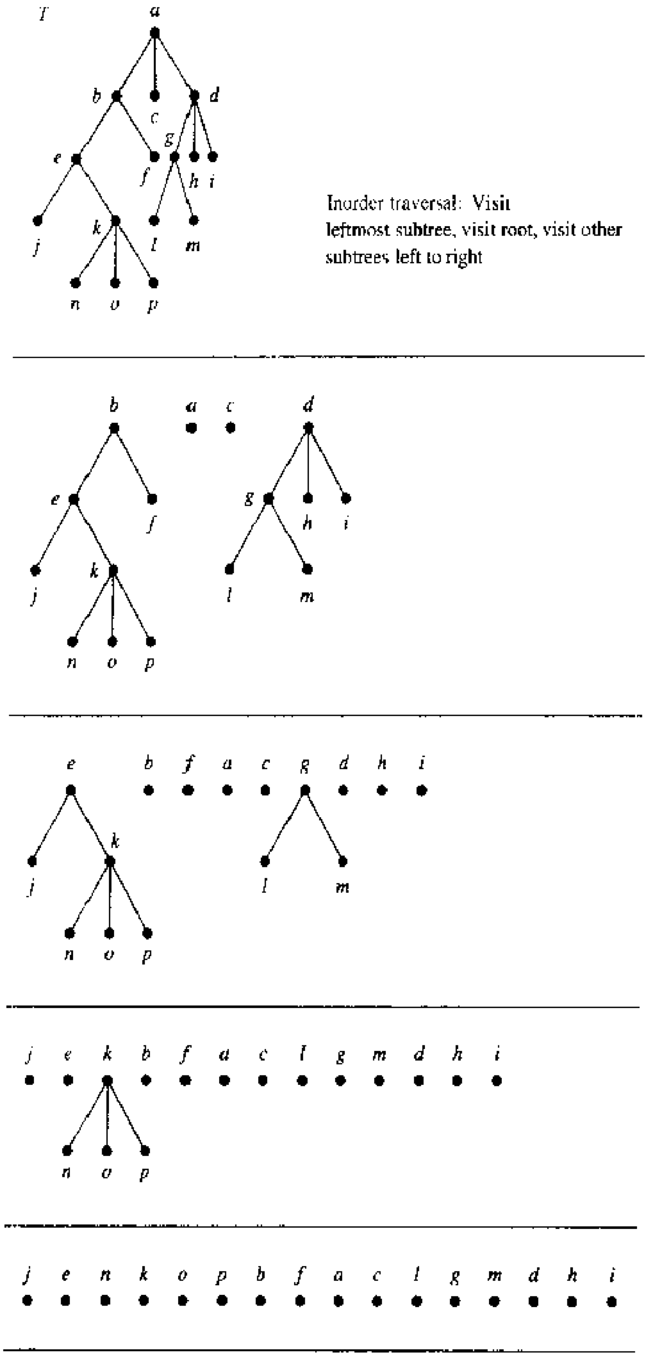
The inorder listing of the subtree with root e is j, followed by the root e, followed by the inorder listing of the subtree with root k. The inorder listing of the subtree with root g is l, g, m. The inorder listing of the subtree with root k is n, k, o, p. Consequently, the inorder listing of the ordered rooted tree is j, e, n, k, o, p, b, f, a, c, l, g, m, d, h, i. ■

The definition of postorder traversal follows.

DEFINITION 3. Let T be an ordered rooted tree with root r. If T consists only of r, then r is the *postorder traversal* of T. Otherwise, suppose that $T_1, T_2, \ldots, T_n$ are the subtrees at r from left to right. The *postorder traversal* begins by traversing T_1 in postorder, then T_2 in postorder, …, then T_n in postorder, and ends by visiting r.

Figure 7 illustrates how postorder traversal is done. The following example illustrates how postorder traversal works.

EXAMPLE 4 In which order does a postorder traversal visit the vertices of the ordered rooted tree T shown in Figure 3?


T
a
b
c
d
e
f
g
h
i
j
k
l
m
n
o
p
b a c d
e b f a c g d h i
j e k b f a c l g m d h i
j e n k o p b f a c l g m d h i

Inorder traversal: Visit leftmost subtree, visit root, visit other subtrees left to right

FIGURE 6 The Inorder Traversal of *T*.

Solution: The steps of the postorder traversal of the ordered rooted tree T are shown in Figure 8. The postorder traversal begins with the postorder traversal of the subtree with root b, the postorder traversal of the subtree with root c, which is just c, the postorder traversal of the subtree with root d, followed by the root a.

The postorder traversal of the subtree with root b begins with the postorder traversal of the subtree with root e, followed by f, followed by the root b. The postorder traversal

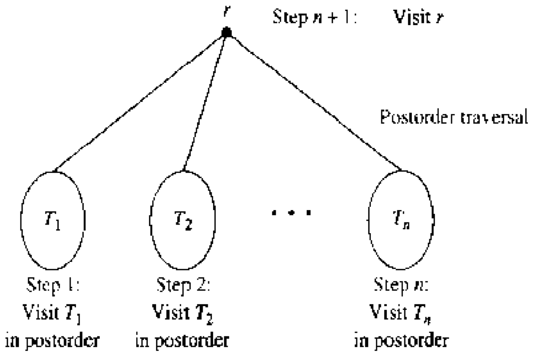

r
Step $n+1$: Visit r
Postorder traversal
T_1
T_2
T_n
Step 1: Visit T_1 in postorder
Step 2: Visit T_2 in postorder
Step n: Visit T_n in postorder

FIGURE 7 **Postorder Traversal.**

of the rooted tree with root d begins with the postorder traversal of the subtree with root g, followed by h, followed by i, followed by the root d.

The postorder traversal of the subtree with root e begins with j, followed by the postorder traversal of the subtree with root k, followed by the root e. The postorder traversal of the subtree with root g is l, m, g. The postorder traversal of the subtree with root k is n, o, p, k. Therefore, the postorder traversal of T is $j, n, o, p, k, e, f, b, c, l, m, g, h, i, d, a$. ■

There are easy ways to list the vertices of an ordered rooted tree in preorder, inorder, and postorder. To do this, first draw a curve around the ordered rooted tree starting at the root, moving along the edges as shown in the example in Figure 9. We can list the vertices in preorder by listing each vertex the first time this curve passes it. We can list the vertices in inorder by listing a leaf the first time the curve passes it and listing each internal vertex the second time the curve passes it. We can list the vertices in postorder by listing a vertex the last time it is passed on the way back up to its parent. When this is done in the rooted tree in Figure 9, it follows that the preorder traversal gives $a, b, d, h, e, i, j, c, f, g, k$, the inorder traversal gives $h, d, b, i, e, j, a, f, c, k, g$, and the postorder traversal gives $h, d, i, j, e, b, f, k, g, c, a$.

Algorithms for traversing ordered rooted trees in preorder, inorder, or postorder are most easily expressed recursively.

ALGORITHM 1 **Preorder Traversal.**

```
procedure preorder(T: ordered rooted tree)
r := root of T
list r
for each child c of r from left to right
begin
  T(c) := subtree with c as its root
  preorder(T(c))
end
```

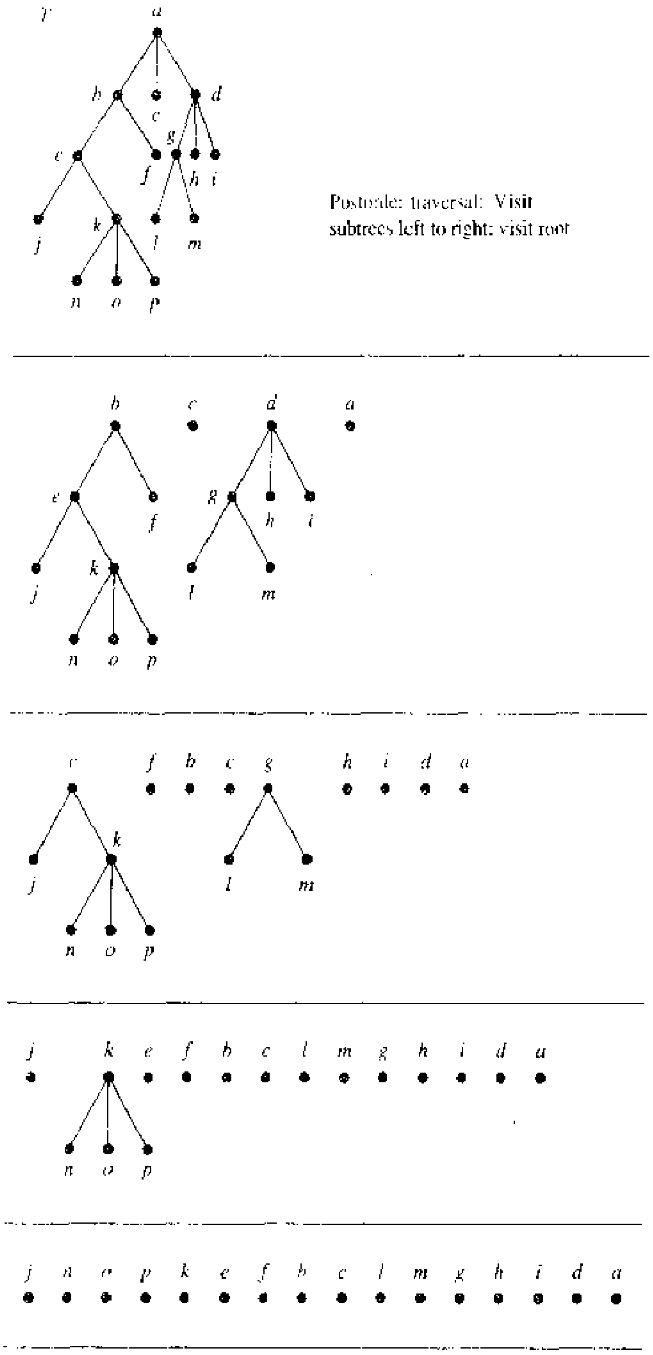

T
Postorder traversal: Visit subtrees left to right; visit root
j n o p k e f b c l m g h i d a

FIGURE 8 The Postorder Traversal of *T*.

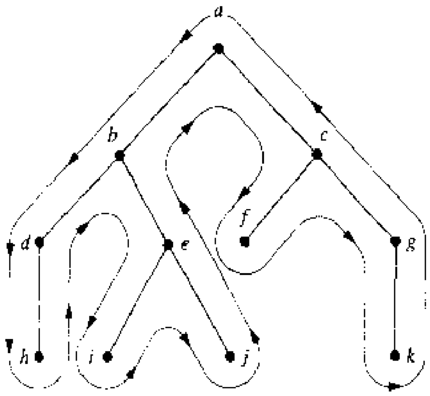


FIGURE 9 A Shortcut for Traversing an Ordered Rooted Tree in Preorder, Inorder, and Postorder.

ALGORITHM 2 **Inorder Traversal.**

```
procedure inorder(T: ordered rooted tree)
r := root of T
if r is a leaf then list r
else
begin
  l := first child of r from left to right
  T(l) := subtree with l as its root
  inorder(T(l))
  list r
  for each child c of r except for l from left to right
    T(c) := subtree with c as its root
    inorder(T(c))
end
```

ALGORITHM 3 **Postorder Traversal.**

```
procedure postorder(T: ordered rooted tree)
r := root of T
for each child c of r from left to right
begin
  T(c) := subtree with c as its root
  postorder(T(c))
end
list r
```

INFIX, PREFIX, AND POSTFIX NOTATION

We can represent complicated expressions, such as compound propositions, combinations of sets, and arithmetic expressions using ordered rooted trees. For instance, consider the representation of an arithmetic expression involving the operators + (addition), − (subtraction), * (multiplication), / (division), and ↑ (exponentiation). We will use parentheses to indicate the order of the operations. An ordered rooted tree can be used to represent such expressions, where the internal vertices represent operations, and the leaves represent the variables or numbers. Each operation operates on its left and right subtrees (in that order).

EXAMPLE 5 What is the ordered rooted tree that represents the expression $((x + y) \uparrow 2) + ((x - 4)/3)$?

Solution: The binary tree for this expression can be built from the bottom up. First, a subtree for the expression $x + y$ is constructed. Then this is incorporated as part of the larger subtree representing $(x + y) \uparrow 2$. Also, a subtree for $x - 4$ is constructed, and then this is incorporated into a subtree representing $(x - 4)/3$. Finally the subtrees representing $(x + y) \uparrow 2$ and $(x - 4)/3$ are combined to form the ordered rooted tree representing $((x + y) \uparrow 2) + ((x - 4)/3)$. These steps are shown in Figure 10. ■

An inorder traversal of the binary tree representing an expression produces the original expression with the elements and operations in the same order as they originally occurred, except for unary operations, which instead immediately follow their operands. For instance, infix traversals of the binary trees in Figure 11, which represent the expressions $(x + y)/(x + 3)$, $(x + (y/x)) + 3$, and $x + (y/(x + 3))$, all lead to the infix expression $x + y/x + 3$. To make such expressions unambiguous it is necessary to include parentheses in the inorder traversal whenever we encounter an operation. The fully parenthesized expression obtained in this way is said to be in **infix form.**

We obtain the **prefix form** of an expression when we traverse its rooted tree in preorder. Expressions written in prefix form are said to be in **Polish notation,** which is named after the logician Jan Łukasiewicz (who was actually Ukrainian and not Polish). An expression in prefix notation (where each operation has a specified number of

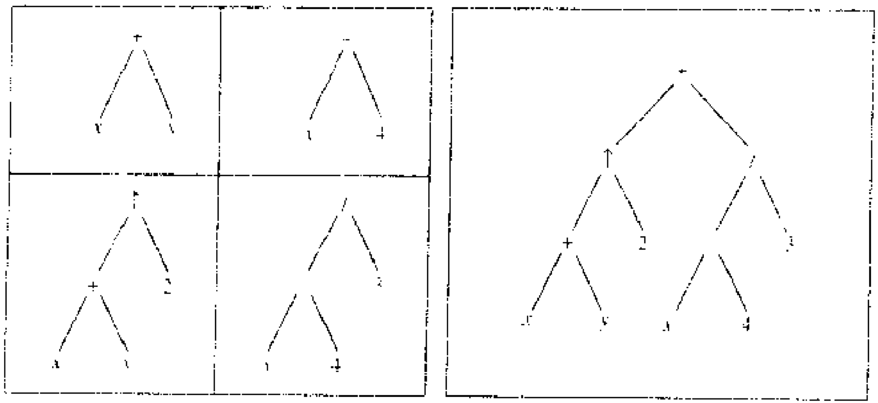


FIGURE 10 A Binary Tree Representing $((x + y) \uparrow 2) + ((x - 4)/3)$.

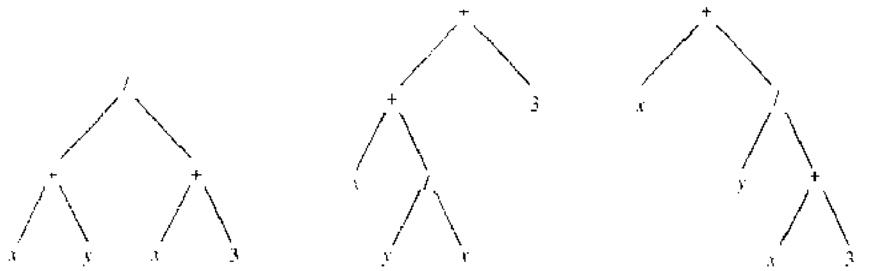


FIGURE 11 **Rooted Trees Representing $(x + y)/(x + 3)$, $(x + (y/x)) + 3$, and $x + (y/(x + 3))$.**

operands), is unambiguous, so no parentheses are needed in such an expression. The verification of this is left as an exercise for the reader.

EXAMPLE 6 What is the prefix form for $((x + y) \uparrow 2) + ((x - 4)/3)$?

Solution: We obtain the prefix form for this expression by traversing the binary tree that represents it, shown in Figure 10. This produces $+ \uparrow + x\ y\ 2 / - x\ 4\ 3$. ■

In the prefix form of an expression, a binary operator, such as +, precedes its two operands. Hence, we can evaluate an expression in prefix form by working from right to left. When we encounter an operator, we perform the corresponding operation with the two operands immediately to the right of this operand. Also, whenever an operation is performed, we consider the result a new operand.

EXAMPLE 7 What is the value of the prefix expression $+ - * 2\ 3\ 5 / \uparrow 2\ 3\ 4$?

Solution: The steps used to evaluate this expression by working right to left, and performing operations using the operands on the right, are shown in Figure 12. The value of this expression is 3. ■

We obtain the **postfix form** of an expression by traversing its binary tree in postorder. Expressions written in postfix form are said to be in **reverse Polish notation.** Expressions in reverse Polish notation are unambiguous, so parentheses are not needed. The verification of this is left to the reader.

EXAMPLE 8 What is the postfix form of the expression $((x + y) \uparrow 2) + ((x - 4)/3)$?

---

web

Jan Lukasiewicz (1878–1956). Jan Lukasiewicz, born in Lwów, studied and began his professional career at the University of Lwów. He later moved to a professorship in Warsaw. After World War II, he was appointed to a position at the Royal Irish Academy in Dublin. Lukasiewicz worked in the area of many-valued logic; his 1921 paper on a three-valued logic was an important contribution to this subject. Nevertheless, he is best known in the mathematical community for his introduction of parenthesis-free notation, now called Polish notation.

$$+ \quad - \quad * \quad 2 \quad 3 \quad 5 \quad / \quad \underbrace{\uparrow \quad 2 \quad 3}_{2 \uparrow 3 = 8} \quad 4$$

$$+ \quad - \quad * \quad 2 \quad 3 \quad 5 \quad \underbrace{/ \quad 8 \quad 4}_{8/4 = 2}$$

$$+ \quad - \quad \underbrace{* \quad 2 \quad 3}_{2 * 3 = 6} \quad 5 \quad 2$$

$$+ \quad \underbrace{- \quad 6 \quad 5}_{6 - 5 = 1} \quad 2$$

$$\underbrace{+ \quad 1 \quad 2}_{1 + 2 = 3}$$

Value of expression: 3

FIGURE 12 **Evaluating a Prefix Expression.**

Solution: The postfix form of the expression is obtained by carrying out a postorder traversal of the binary tree for this expression, shown in Figure 10. This produces the postfix expression: $x\ y + 2 \uparrow x\ 4 - 3\ / +$. ■

In the postfix form of an expression, a binary operator follows its two operands. So, to evaluate an expression from its postfix form, work from left to right, carrying out operations whenever an operator follows two operands. After an operation is carried out, the result of this operation becomes a new operand.

EXAMPLE 9 What is the value of the postfix expression $7\ 2\ 3 * - 4 \uparrow 9\ 3\ / +$?

Solution: The steps used to evaluate this expression by starting at the left and carrying out operations when two operands are followed by an operator are shown in Figure 13. The value of this expression is 4. ■

Rooted trees can be used to represent other types of expressions, such as those representing compound propositions and combinations of sets. In these examples unary operators, such as the negation of a proposition, occur. To represent such operators and their operands, a vertex representing the operator and a child of this vertex representing the operand are used.

EXAMPLE 10 Find the ordered rooted tree representing the compound proposition $(\neg(p \wedge q)) \leftrightarrow (\neg p \vee \neg q)$. Then use this rooted tree to find the prefix, postfix, and infix forms of this expression.

$$7\ \underbrace{2\ \ 3\ \ *}_{2*3=6}\ -\ 4\ \uparrow\ 9\ 3\ /\ +$$

$$\underbrace{7\ \ 6\ \ -}_{7-6=1}\ 4\ \uparrow\ 9\ 3\ /\ +$$

$$\underbrace{1\ \ 4\ \ \uparrow}_{1^4=1}\ 9\ 3\ /\ +$$

$$1\ \underbrace{9\ \ 3\ \ /}_{9/3=3}\ +$$

$$\underbrace{1\ \ 3\ \ +}_{1+3=4}$$

FIGURE 13 Evaluating a Postfix Expression.

Solution: The rooted tree for this compound proposition is constructed from the bottom up. First, subtrees for $\neg p$ and $\neg q$ are formed (where $\neg$ is considered a unary operator). Also, a subtree for $p \wedge q$ is formed. Then subtrees for $\neg(p \wedge q)$ and $(\neg p) \vee (\neg q)$ are constructed. Finally, these two subtrees are used to form the final rooted tree. The steps of this procedure are shown in Figure 14.

The prefix, postfix, and infix forms of this expression are found by traversing this rooted tree in preorder, postorder, and inorder (including parentheses), respectively. These traversals give $\leftrightarrow \neg \wedge pq \vee \neg p \neg q$, $pq \wedge \neg p \neg q \neg \vee \leftrightarrow$, and $(\neg(p \wedge q)) \leftrightarrow ((\neg p) \vee (\neg q))$, respectively. ∎

Because prefix and postfix expressions are unambiguous and because they can easily be evaluated without scanning back and forth, they are used extensively in computer science. Such expressions are especially useful in the construction of compilers.

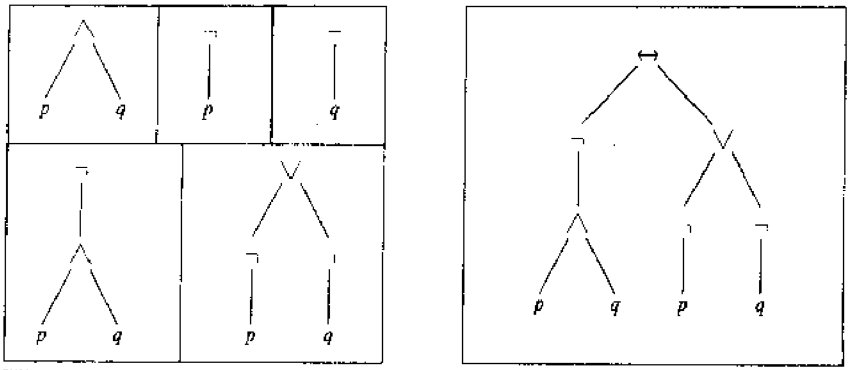
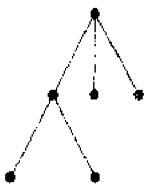


FIGURE 14 Constructing the Rooted Tree for a Compound Proposition.

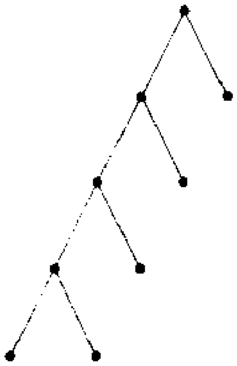
Exercises

In Exercises 1–3 construct the universal address system for the given ordered rooted tree. Then use this to order its vertices using the lexicographic order of their labels.

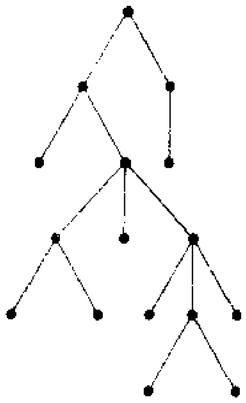
1.



2.



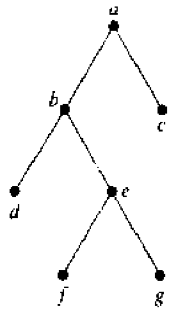
3.



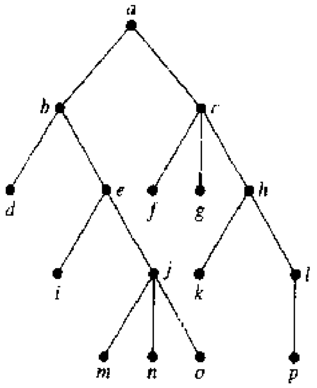
4. Suppose that the address of the vertex v in the ordered rooted tree T is 3.4.5.2.4.
   a) At what level is v?
   b) What is the address of the parent of v?
   c) What is the least number of siblings v can have?
   d) What is the smallest possible number of vertices in T if v has this address?
   e) Find the other addresses that must occur.
5. Suppose that the vertex with the largest address in an ordered rooted tree T has address 2.3.4.3.1. Is it possible to determine the number of vertices in T?
6. Can the leaves of an ordered rooted tree have the following list of universal addresses? If so, construct such an ordered rooted tree.
   a) 1.1.1, 1.1.2, 1.2, 2.1.1.1, 2.1.2, 2.1.3, 2.2, 3.1.1, 3.1.2.1, 3.1.2.2, 3.2
   b) 1.1, 1.2.1, 1.2.2, 1.2.3, 2.1, 2.2.1, 2.3.1, 2.3.2, 2.4.2.1, 2.4.2.2, 3.1, 3.2.1, 3.2.2
   c) 1.1, 1.2.1, 1.2.2, 1.2.2.1, 1.3, 1.4, 2, 3.1, 3.2, 4.1.1.1

In Exercises 7–9 determine the order in which a preorder traversal visits the vertices of the given ordered rooted tree.

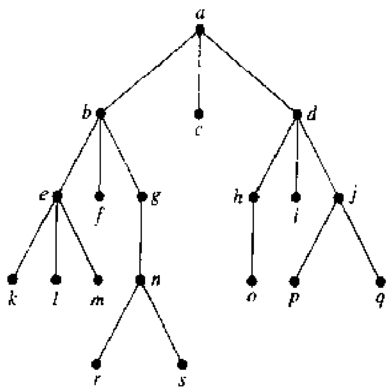
7.



8.



9.



10. In which order are the vertices of the ordered rooted tree in Exercise 7 visited using an inorder traversal?

11. In which order are the vertices of the ordered rooted tree in Exercise 8 visited using an inorder traversal?

12. In which order are the vertices of the ordered rooted tree in Exercise 9 visited using an inorder traversal?

13. In which order are the vertices of the ordered rooted tree in Exercise 7 visited using a postorder traversal?

14. In which order are the vertices of the ordered rooted tree in Exercise 8 visited using a postorder traversal?

15. In which order are the vertices of the ordered rooted tree in Exercise 9 visited using a postorder traversal?

16. Represent the expression $((x+2) \uparrow 3)*(y-(3+x))-5$ using a binary tree.

17. Write the expression in Exercise 16 in
- **a)** prefix notation.
- **b)** postfix notation.
- **c)** infix notation.

18. Represent the expressions $(x + xy) + (x/y)$ and $x + ((xy + x)/y)$ using binary trees.

19. Write the expressions in Exercise 18 in
- **a)** prefix notation.
- **b)** postfix notation.
- **c)** infix notation.

20. Represent the compound propositions $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$ and $(\neg p \wedge (q \leftrightarrow \neg p)) \vee \neg q$ using ordered rooted trees.

21. Write the expressions in Exercise 20 in
- **a)** prefix notation.
- **b)** postfix notation.
- **c)** infix notation.

22. Represent $(A \cap B) - (A \cup (B - A))$ using an ordered rooted tree.

23. Write the expression in Exercise 22 in
- **a)** prefix notation.
- **b)** postfix notation.
- **c)** infix notation.

***24.** In how many ways can the string $\neg p \wedge q \leftrightarrow \neg p \vee \neg q$ be fully parenthesized to yield an infix expression?

***25.** In how many ways can the string $A \cap B - A \cap B - A$ be fully parenthesized to yield an infix expression?

26. Draw the ordered rooted tree corresponding to each of the following arithmetic expressions written in prefix notation. Then write each expression using infix notation.
- **a)** $+ * + - 5\ 3\ 2\ 1\ 4$
- **b)** $\uparrow + 2\ 3 - 5\ 1$
- **c)** $* / 9\ 3 + * 2\ 4 - 7\ 6$

27. What is the value of each of the following prefix expressions?
- **a)** $- * 2 / 8\ 4\ 3$
- **b)** $\uparrow - * 3\ 3 * 4\ 2\ 5$
- **c)** $+ - \uparrow 3\ 2 \uparrow 2\ 3 / 6 - 4\ 2$
- **d)** $* + 3 + 3 \uparrow 3 + 3\ 3\ 3$

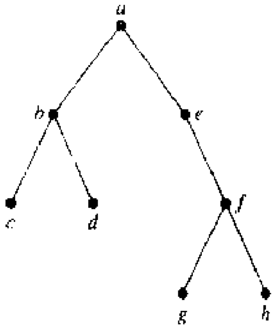
28. What is the value of each of the following postfix expressions?
- **a)** $5\ 2\ 1 - - 3\ 1\ 4 + + *$
- **b)** $9\ 3 / 5 + 7\ 2 - *$
- **c)** $3\ 2 * 2 \uparrow 5\ 3 - 8\ 4 / * -$

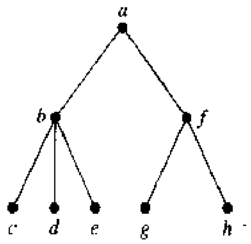
29. Construct the ordered rooted tree whose preorder traversal is *a*, *b*, *f*, *c*, *g*, *h*, *i*, *d*, *e*, *j*, *k*, *l*, where *a* has four children, *c* has three children, *j* has two children, *b* and *e* have one child each, and all other vertices are leaves.

***30.** Show that an ordered rooted tree is uniquely determined when a list of vertices generated by a preorder traversal of the tree and the number of children of each vertex are specified.

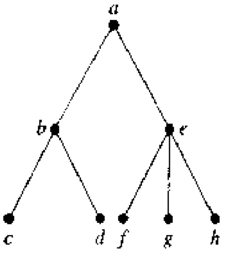
***31.** Show that an ordered rooted tree is uniquely determined when a list of vertices generated by a postorder traversal of the tree and the number of children of each vertex are specified.

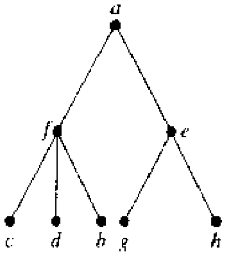
32. Show that preorder traversals of the two ordered rooted trees displayed below produce the same list of vertices. Note that this does not contradict the statement in Exercise 30, since the numbers of children of internal vertices in the two ordered rooted trees differ.





33. Show that postorder traversals of the two ordered rooted trees (on the next page) produce the same list of vertices. Note that this does not contradict the statement in Exercise 31, since the numbers of children of internal vertices in the two ordered rooted trees differ.





Well-formed formulae in prefix notation over a set of symbols and a set of binary operators are defined recursively by the following rules:

(i) if x is a symbol, then x is a well-formed formula in prefix notation;

(ii) if X and Y are well-formed formulae and $*$ is an operator, then $* XY$ is a well-formed formula.

34. Which of the following are well-formed formulae over the symbols $\{x, y, z\}$ and the set of binary operators $\{\times, +, \circ\}$?

a) $\times + + x\ y\ x$

b) $\circ\ x\ y \times x\ z$

c) $\times \circ x\ z \times \times x\ y$

d) $\times + \circ x\ x \circ x\ x\ x$

***35.** Show that any well-formed formula in prefix notation over a set of symbols and a set of binary operators contains exactly one more symbol than the number of operators.

36. Give a definition of well-formed formulae in postfix notation over a set of symbols and a set of binary operators.

37. Give six examples of well-formed formulae with three or more operators in postfix notation over the set of symbols $\{x, y, z\}$ and the set of operators $\{+, \times, \circ\}$.

38. Extend the definition of well-formed formulae in prefix notation to sets of symbols and operators where the operators may not be binary.

8.4

Trees and Sorting

INTRODUCTION

The problem of ordering the elements in a set occurs in many contexts. For instance, to produce a printed telephone directory it is necessary to alphabetize the names of the subscribers.

web Suppose that there is a total ordering of the elements of a set. Initially the elements in a set may be in any order. A **sorting** is a reordering of these elements into a list in which the elements are in increasing order. For instance, sorting the list 7, 2, 1, 4, 5, 9 produces the list 1, 2, 4, 5, 7, 9. Sorting the list d, h, c, a, f (using alphabetical order) produces the list a, c, d, f, h.

A large percentage of computer use is devoted to sorting one thing or another. Hence, much effort has been devoted to the development of efficient sorting algorithms. In this section several sorting algorithms and their computational complexity will be discussed. As will be seen in this section, trees are used to describe sorting algorithms and are used in the analysis of their complexity.

THE COMPLEXITY OF SORTING

Many different sorting algorithms have been developed. To decide whether a particular sorting algorithm is efficient, its complexity is determined. Using trees as models, a lower bound for the worst-case complexity of sorting algorithms can be found.

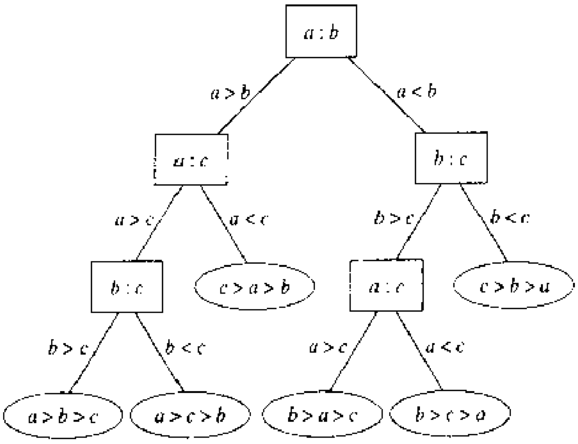


FIGURE 1 **A Decision Tree for Sorting Three Distinct Elements.**

There are $n!$ possible orderings of n elements, since each of the $n!$ permutations of these elements can be the correct order. The sorting algorithms we will study are based on binary comparisons, that is, the comparison of two elements at a time. The result of each such comparison narrows down the set of possible orderings. Thus, a sorting algorithm based on binary comparisons can be represented by a binary decision tree in which each internal vertex represents a comparison of two elements. Each leaf represents one of the $n!$ permutations of n elements.

EXAMPLE 1 We display in Figure 1 a decision tree that orders the elements of the list a, b, c. ■

The complexity of a sort based on binary comparisons is measured in terms of the number of such comparisons used. The most binary comparisons ever needed to sort a list with n elements gives the worst-case performance of the algorithm. The most comparisons used equals the longest path length in the decision tree representing the sorting procedure. In other words, the most comparisons ever needed is equal to the height of the decision tree. Since the height of a binary tree with $n!$ leaves is at least $\lceil \log n! \rceil$ (using Corollary 1 in Section 8.1), at least $\lceil \log n! \rceil$ comparisons are needed, as stated in Theorem 1.

THEOREM 1 **A sorting algorithm based on binary comparisons requires at least $\lceil \log n! \rceil$ comparisons.**

By Example 5 of Section 1.8, it follows that $\lceil \log n! \rceil$ is $O(n \log n)$. In fact, greater than $(n \log n)/4$ for $n > 4$ (see Exercise 18), it follows that no sorting algorithm that uses comparisons as the method of sorting can have worst-case time complexity that is better than $O(n \log n)$. Consequently, a sorting algorithm is as efficient as possible (in the sense of a big-O estimate of time complexity) if it has $O(n \log n)$ time complexity.

THE BUBBLE SORT

web The **bubble sort** is one of the simplest sorting algorithms but not one of the most efficient. It puts a list into increasing order by successively comparing adjacent elements, interchanging them if they are in the wrong order. To carry out the bubble sort, we perform the basic operation, that is, interchanging a larger element with a smaller one following it, starting at the beginning of the list, for a full pass. We iterate this procedure until the sort is complete. We can imagine the elements in the list placed in a column. In the bubble sort, the smaller elements "bubble" to the top as they are interchanged with larger elements. The larger elements "sink" to the bottom. This is illustrated in the following example.

EXAMPLE 2 Use the bubble sort to put 3, 2, 4, 1, 5 into increasing order.

Solution: Begin by comparing the first two elements, 3 and 2. Since $3 > 2$, interchange 3 and 2, producing the list 2, 3, 4, 1, 5. Since $3 < 4$, continue by comparing 4 and 1. Since $4 > 1$, interchange 1 and 4, producing the list 2, 3, 1, 4, 5. Since $4 < 5$, the first pass is complete. The first pass guarantees that the largest element, 5, is in the correct position.

The second pass begins by comparing 2 and 3. Since these are in the correct order, 3 and 1 are compared. Since $3 > 1$, these numbers are interchanged, producing 2, 1, 3, 4, 5. Since $3 < 4$, these numbers are in the correct order. It is not necessary to do any more comparisons for this pass because 5 is already in the correct position. The second pass guarantees that the two largest elements, 4 and 5, are in their correct positions.

The third pass begins by comparing 2 and 1. These are interchanged since $2 > 1$, producing 1, 2, 3, 4, 5. Because $2 < 3$, these two elements are in the correct order. It is not necessary to do any more comparisons for this pass because 4 and 5 are already in the correct positions. The third pass guarantees that the three largest elements, 3, 4, and 5, are in their correct positions.

The fourth pass consists of one comparison, namely, the comparison of 1 and 2. Since $1 < 2$, these elements are in the correct order. This completes the bubble sort.

The steps of this algorithm are illustrated in Figure 2. ■

A pseudocode description of the bubble sort is given in Algorithm 1.

How efficient is the bubble sort? Since $n - i$ comparisons are used during the ith pass, the total number of comparisons used in a bubble sort of a list of n elements is

$$(n-1) + (n-2) + \cdots + 2 + 1.$$

This is the sum of the $n - 1$ smallest integers. From Example 9 of Section 3.2, this equals $(n-1)n/2$. Consequently, the bubble sort uses $n(n-1)/2$ comparisons to order a list of n elements. (Note that the bubble sort always uses this many comparisons, since it continues even if the list becomes completely sorted at some intermediate step.) Hence, the bubble sort algorithm has worst-case complexity $O(n^2)$. Since for every

ALGORITHM 1 The Bubble Sort.

```
procedure bubblesort($a_1, \ldots, a_n$)
for $i := 1$ to $n - 1$
begin
  for $j := 1$ to $n - i$
    if $a_j > a_{j+1}$ then interchange $a_j$ and $a_{j+1}$
end
{$a_1, \ldots, a_n$ is in increasing order}
```

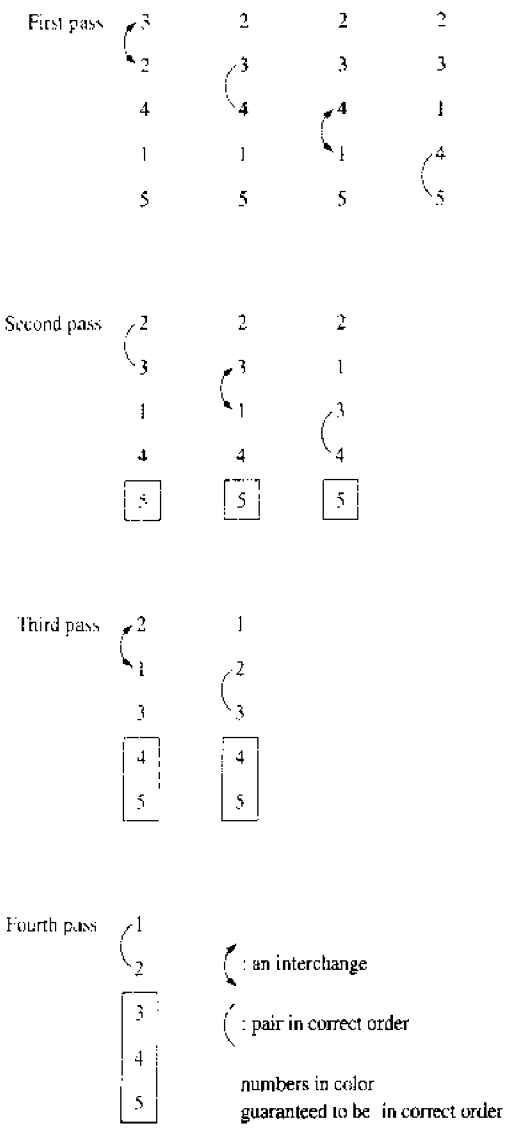

First pass
3
2
4
1
5
Second pass
Third pass
Fourth pass
: an interchange
: pair in correct order
numbers in color
guaranteed to be in correct order

FIGURE 2 **The Steps of a Bubble Sort.**

positive real number c, $n(n-1)/2 > cn \log n$ for some sufficiently large positive integer n, it follows that the bubble sort does not have $O(n \log n)$ worst-case time complexity. We need to find another algorithm in order to achieve this optimal estimate of worst-case complexity.

THE MERGE SORT

web Many different sorting algorithms achieve the best possible worst-case complexity for a sorting algorithm, namely, $O(n \log n)$ comparisons to sort n elements. We will describe one of these algorithms, called the **merge sort** algorithm, here. We will demonstrate how the merge sort algorithm works with an example before describing it in generality.

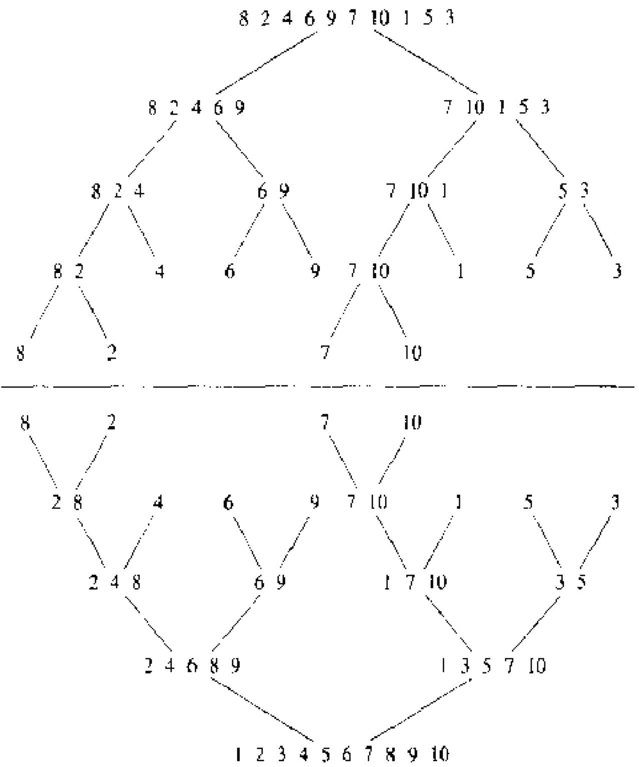


FIGURE 3 **The Merge Sort of 8, 2, 4, 6, 9, 7, 10, 1, 5, 3.**

EXAMPLE 3 We will sort the list 8, 2, 4, 6, 9, 7, 10, 1, 5, 3 using the merge sort. A merge sort begins by splitting the list into individual elements by successively splitting lists in two. The progression of sublists for this example is represented with the balanced binary tree of height 4 shown in the upper half of Figure 3.

Sorting is done by successively merging pairs of lists. At the first stage, pairs of individual elements are merged into lists of length two in increasing order. Then successive merges of pairs of lists are performed until the entire list is put into increasing order. The succession of merged lists in increasing order is represented by the balanced binary tree of height 4 shown in the lower half of Figure 3 (note that this tree is displayed "upside down"). ■

In general, a merge sort proceeds by iteratively splitting lists into two sublists of equal length (or where one sublist has one more element than the other) until each sublist contains one element. This succession of sublists can be represented by a balanced binary tree. The procedure continues by successively merging pairs of lists, where both lists are in increasing order, into a larger list with elements in increasing order, until the original list is put into increasing order. The succession of merged lists can be represented by a balanced binary tree.

We can also describe the merge sort recursively. To do a merge sort, we split a list into two sublists of equal, or approximately equal, size, sorting each sublist using the merge sort algorithm, and then merging the two lists. We leave it for the reader to give a complete specification of the recursive version of the merge sort.

TABLE 1 **Merging the Sorted Lists 2, 3, 5, 6; and 1, 4.**

First List	*Second List*	*Merged List*	*Comparison*
2 3 5 6	1 4		$1 < 2$
2 3 5 6	4	1	$2 < 4$
3 5 6	4	1 2	$3 < 4$
5 6	4	1 2 3	$4 < 5$
5 6		1 2 3 4	
		1 2 3 4 5 6	

An efficient algorithm for merging two ordered lists into a larger ordered list is needed to implement the merge sort. We will now describe such a procedure.

EXAMPLE 4 We will describe how to merge the two lists 2, 3, 5, 6 and 1, 4. Table 1 illustrates the steps we use.

First, compare the smallest elements in the two lists, 2 and 1, respectively. Since 1 is the smaller, put it at the beginning of the merged list and remove it from the second list. At this stage, the first list is 2, 3, 5, 6, the second is 4, and the combined list is 1.

Next, compare 2 and 4, the smallest elements of the two lists. Since 2 is the smaller, add it to the combined list and remove it from the first list. At this stage the first list is 3, 5, 6, the second is 4, and the combined list is 1, 2.

Continue by comparing 3 and 4, the smallest elements of their respective lists. Since 3 is the smaller of these two elements, add it to the combined list and remove it from the first list. At this stage the first list is 5, 6, and the second is 4. The combined list is 1, 2, 3.

Then compare 5 and 4, the smallest elements in the two lists. Since 4 is the smaller of these two elements, add it to the combined list and remove it from the second list. At this stage the first list is 5, 6, the second list is empty, and the combined list is 1, 2, 3, 4.

Finally, since the second list is empty, all elements of the first list can be appended to the end of the combined list in the order they occur in the first list. This produces the ordered list 1, 2, 3, 4, 5, 6. ■

We will now consider the general problem of merging two ordered lists L_1 and L_2, into an ordered list L. We can use the following procedure. Start with an empty list L. Compare the smallest elements of the two lists. Put the smaller of these two elements at the end of L, and remove it from the list it was in. Next, if one of L_1 and L_2 is empty, append the other (nonempty) list to L, which completes the merging. If neither L_1 nor L_2 is empty, repeat this process. Algorithm 2 gives a pseudocode description of this procedure.

We will need estimates for the number of comparisons used to merge two ordered lists in the analysis of the merge sort. We can easily obtain such an estimate for Algorithm 2. Each time a comparison of an element from L_1 and an element from L_2 is made, an additional element is added to the merged list L. However, when either L_1 or L_2 is empty, no more comparisons are needed. Hence, Algorithm 2 is least efficient

ALGORITHM 2 **Merging Two Lists.**

```
procedure merge($L_1, L_2$ : lists)
$L$ := empty list
while $L_1$ and $L_2$ are both nonempty
begin
  remove smaller of first element of $L_1$ and $L_2$ from the list it is
      in and put it at the end of $L$
  if removal of this element makes one list empty then remove
      all elements from the other list and append them to $L$
end {$L$ is the merged list with elements in increasing order}
```

when $m + n - 2$ comparisons are carried out, where m and n are the number of elements in L_1 and L_2, respectively, leaving one element in each of L_1 and L_2. The next comparison will be the last one needed, because it will make one of these lists empty. Hence, Algorithm 2 uses no more than $m + n - 1$ comparisons. The following lemma summarizes this estimate.

LEMMA 1 **Two sorted lists with m elements and n elements can be merged into a sorted list using no more than $m + n - 1$ comparisons.**

Sometimes two sorted lists of length m and n can be merged using far fewer than $m + n - 1$ comparisons. For instance, when $m = 1$, a binary search procedure can be applied to put the one element in the first list into the second list. This requires only $\lceil \log n \rceil$ comparisons, which is much smaller than $m + n - 1 = n$, for $m = 1$. On the other hand, for some values of m and n, Lemma 1 gives the best possible bound. That is, there are lists with m and n elements that cannot be merged using fewer than $m + n - 1$ comparisons. (See Exercise 7 at the end of this section.)

We can now analyze the complexity of the merge sort. Instead of studying the general problem, we will assume that n, the number of elements in the list, is a power of 2, say 2^m. This will make the analysis less complicated, but when this is not the case, various modifications can be applied that will yield the same estimate.

At the first stage of the splitting procedure, the list is split into two sublists, of 2^{m-1} elements each, at level 1 of the tree generated by the splitting. This process continues, splitting the two sublists with 2^{m-1} elements into four sublists of 2^{m-2} elements each at level 2, and so on. In general, there are 2^{k-1} lists at level $k - 1$, each with 2^{m-k+1} elements. These lists at level $k - 1$ are split into 2^k lists at level k, each with 2^{m-k} elements. At the end of this process, we have 2^m lists each with one element at level m.

We start merging by combining pairs of the 2^m lists of one element into 2^{m-1} lists, at level $m - 1$, each with two elements. To do this, 2^{m-1} pairs of lists with one element each are merged. The merger of each pair requires exactly one comparison.

The procedure continues, so that at level k ($k = m, m-1, m-2, \ldots, 3, 2, 1$), 2^k lists each with 2^{m-k} elements are merged into 2^{k-1} lists, each with 2^{m-k+1} elements, at level $k-1$. To do this a total of 2^{k-1} mergers of two lists, each with 2^{m-k} elements, are needed. But, by Lemma 1, each of these mergers can be carried out using at most $2^{m-k} + 2^{m-k} - 1 = 2^{m-k+1} - 1$ comparisons. Hence, going from level k to $k - 1$ can be accomplished using at most $2^{k-1}(2^{m-k+1} - 1)$ comparisons. Summing all these estimates shows that the number of comparisons required for the merge sort is at most

$$\sum_{k=1}^{m} 2^{k-1}(2^{m-k+1}-1) = \sum_{k=1}^{m} 2^{m} - \sum_{k=1}^{m} 2^{k-1} = m2^{m} - (2^{m}-1) = n \log n - n + 1,$$

since $m = \log n$ and $n = 2^m$. (We evaluated $\sum_{k=1}^{m} 2^m$ by noting that it is the sum of m identical terms, each equal to 2^m. We evaluated $\sum_{k=1}^{m} 2^{k-1}$ using the formula for the sum of the terms of a geometric progression from Example 6 of Section 3.2.)

This analysis shows that the merge sort achieves the best possible big-O estimate for the number of comparisons needed by sorting algorithms, as stated in the following theorem.

THEOREM 2 **The number of comparisons needed to merge-sort a list with n elements is $O(n \log n)$.**

We describe another efficient algorithm, the quick sort, in the exercises.

Exercises

1. Use a bubble sort to sort 3, 1, 5, 7, 4, showing the lists obtained at each step.
2. Use a bubble sort to sort *d, f, k, m, a, b*, showing the lists obtained at each step.
*3. Adapt the bubble sort algorithm so that it stops when no interchanges are required. Express this more efficient version of the algorithm in pseudocode.
4. Use a merge sort to sort 4, 3, 2, 5, 1, 8, 7, 6. Show all the steps used by the algorithm.
5. Use a merge sort to sort *b, d, a, f, g, h, z, p, o, k*. Show all the steps used by the algorithm.
6. How many comparisons are required to merge the following pairs of lists using Algorithm 2?
   a) 1, 3, 5, 7, 9; 2, 4, 6, 8, 10
   b) 1, 2, 3, 4, 5; 6, 7, 8, 9, 10
   c) 1, 5, 6, 7, 8; 2, 3, 4, 9, 10
7. Show that there are lists with m elements and n elements such that they cannot be merged into one sorted list using Algorithm 2 with fewer than $m + n - 1$ comparisons.
*8. What is the least number of comparisons needed to merge any two lists in increasing order into one list in increasing order when the number of elements in the two lists are
   a) 1, 4? b) 2, 4?
   c) 3, 4? d) 4, 4?

The **selection sort** begins by finding the least element in the list. This element is moved to the front. Then the least element among the remaining elements is found and put into the second position. This procedure is repeated until the entire list has been sorted.

9. Sort the following lists using the selection sort.
   a) 3, 5, 4, 1, 2 b) 5, 4, 3, 2, 1 c) 1, 2, 3, 4, 5
10. Write the selection sort algorithm in pseudocode.
11. How many comparisons are used to perform a selection sort of n items?

The **quick sort** is an efficient algorithm. To sort $a_1, a_2, \ldots, a_n$, this algorithm begins by taking the first element a_1 and forming two sublists, the first containing those elements that are less than a_1, in the order they arise, and the second containing those elements greater than a_1, in the order they arise. Then a_1 is put at the end of the first sublist. This procedure is repeated recursively for each sublist, until all sublists contain one item. The ordered list of n items is obtained by combining the sublists of one item in the order they occur.

12. Sort 3, 5, 7, 8, 1, 9, 2, 4, 6 using the quick sort.
13. Let $a_1, a_2, \ldots, a_n$ be a list of n distinct real numbers. How many comparisons are needed to form two sublists from this list, the first containing elements less than a_1 and the second containing elements greater than a_1?
14. Describe the quick sort algorithm using pseudocode.
15. What is the largest number of comparisons needed to order a list of four elements using the quick sort algorithm?
16. What is the least number of comparisons needed to order a list of four elements using the quick sort algorithm?
17. Determine the worst-case complexity of the quick sort algorithm in terms of the number of comparisons used.
*18. Show that $\log n!$ is greater than $(n \log n)/4$ for $n > 4$. [*Hint:* Begin with the inequality $n! > n(n-1)(n-2)\cdots\lceil n/2\rceil$.]
*19. Write the merge sort algorithm in pseudocode.

8.5

Spanning Trees

INTRODUCTION

Consider the system of roads in Maine represented by the simple graph shown in Figure 1(a). The only way the roads can be kept open in the winter is by frequently plowing them. The highway department wants to plow the fewest roads so that there will always be cleared roads connecting any two towns. How can this be done?

At least five roads must be plowed to ensure that there is a path between any two towns. Figure 1(b) shows one such set of roads. Note that the subgraph representing these roads is a tree, since it is connected and contains six vertices and five edges.

This problem was solved with a connected subgraph with the minimum number of edges containing all vertices of the original simple graph. Such a graph must be a tree.

DEFINITION 1. **Let G be a simple graph. A *spanning tree* of G is a subgraph of G that is a tree containing every vertex of G.**

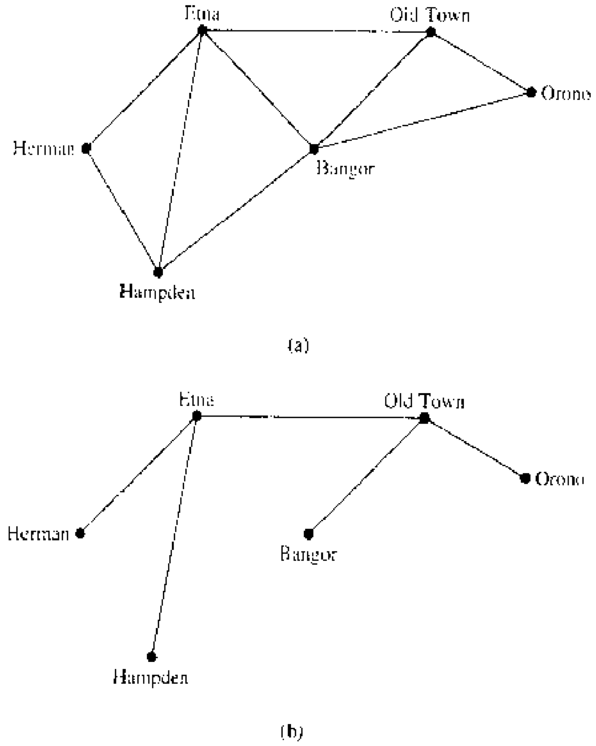


FIGURE 1 **(a) A Road System and (b) A Set of Roads to Plow.**

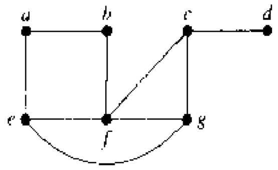


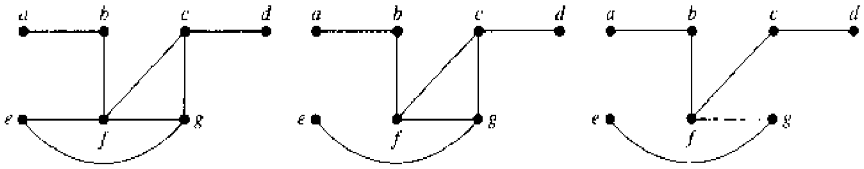
FIGURE 2 The Simple Graph G.

A simple graph with a spanning tree must be connected, since there is a path in the spanning tree between any two vertices. The converse is also true; that is, every connected simple graph has a spanning tree. We will give an example before proving this result.

EXAMPLE 1 Find a spanning tree of the simple graph G shown in Figure 2.

Solution: The graph G is connected, but it is not a tree because it contains simple circuits. Remove the edge $\{a, e\}$. This eliminates one simple circuit, and the resulting subgraph is still connected and still contains every vertex of G. Next remove the edge $\{e, f\}$ to eliminate a second simple circuit. Finally, remove edge $\{c, g\}$ to produce a simple graph with no simple circuits. This subgraph is a spanning tree, since it is a tree that contains every vertex of G. The sequence of edge removals used to produce the spanning tree is illustrated in Figure 3.

The tree shown in Figure 3 is not the only spanning tree of G. For instance, each of the trees shown in Figure 4 is a spanning tree of G. ■



Edge removed: $\{a, e\}$ $\{e, f\}$ $\{c, g\}$

(a) (b) (c)

FIGURE 3 Producing a Spanning Tree for G by Removing Edges That Form Simple Circuits.

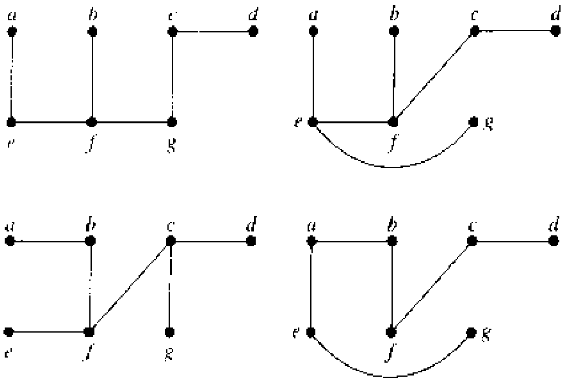


FIGURE 4 Spanning Trees of G.

THEOREM 1 **A simple graph is connected if and only if it has a spanning tree.**

Proof: First, suppose that a simple graph G has a spanning tree T. T contains every vertex of G. Furthermore, there is a path in T between any two of its vertices. Since T is a subgraph of G, there is a path in G between any two of its vertices. Hence, G is connected.

Now suppose that G is connected. If G is not a tree, it must contain a simple circuit. Remove an edge from one of these simple circuits. The resulting subgraph has one fewer edge but still contains all the vertices of G and is connected. If this subgraph is not a tree, it has a simple circuit; so as before, remove an edge that is in a simple circuit. Repeat this process until no simple circuits remain. This is possible because there are only a finite number of edges in the graph. The process terminates when no simple circuits remain. A tree is produced since the graph stays connected as edges are removed. This tree is a spanning tree since it contains every vertex of G. □

Spanning trees are important in data networking, as the following example shows.

EXAMPLE 2 *web* *IP Multicasting* Spanning trees play an important role in multicasting over Internet Protocol (IP) networks. To send data from a source computer to multiple receiving computers, each of which is a subnetwork, data could be sent separately to each computer. This type of networking, called unicasting, is inefficient, since many copies of the same data are transmitted over the network. To make the transmission of data to multiple receiving computers more efficient, IP multicasting is used. With IP multicasting, a computer sends a single copy of data over the network, and as data reaches intermediate routers the data are forwarded to one or more other routers so that ultimately all receiving computers in their various subnetworks receive these data. (Routers are computers that are dedicated to forwarding IP datagrams between subnetworks in a network. In multicasting, routers use Class D addresses, each representing a session that receiving computers may join; see Example 8 in Section 4.3.)

For data to reach receiving computers as quickly as possible, there should be no loops (which in graph theory terminology are circuits or cycles) in the path that data take through the network. That is, once data have reached a particular router, data should never return to this router. To avoid loops, the multicast routers use network algorithms to construct a spanning tree in the graph that has the multicast source, the routers, and the subnetworks containing receiving computers as vertices, with edges representing the links between computers and/or routers. The root of this spanning tree is the multicast source. The subnetworks containing receiving computers are leaves of the tree. (Note that subnetworks not containing receiving stations are not included in the graph.) This is illustrated in Figure 5. ■

ALGORITHMS FOR CONSTRUCTING SPANNING TREES

web The proof of Theorem 1 gives an algorithm for finding spanning trees by removing edges from simple circuits. This algorithm is inefficient, since it requires that simple circuits be identified. Instead of constructing spanning trees by removing edges, spanning trees can be built up by successively adding edges. Two algorithms based on this principle will be presented here.

We can build a spanning tree for a connected simple graph using a **depth-first search.** We will form a rooted tree, and the spanning tree will be the underlying

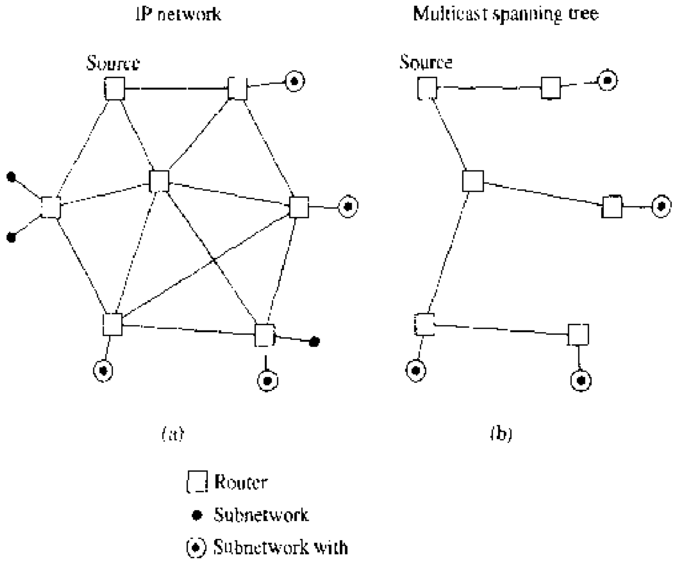


FIGURE 5 **A Multicase Spanning Tree.**

undirected graph of this rooted tree. Arbitrarily choose a vertex of the graph as the root. Form a path starting at this vertex by successively adding edges, where each new edge is incident with the last vertex in the path and a vertex not already in the path. Continue adding edges to this path as long as possible. If the path goes through all vertices of the graph, the tree consisting of this path is a spanning tree. However, if the path does not go through all vertices, more edges must be added. Move back to the next to last vertex in the path, and, if possible, form a new path starting at this vertex passing through vertices that were not already visited. If this cannot be done, move back another vertex in the path, that is, two vertices back in the path, and try again. Repeat this procedure, beginning at the last vertex visited, moving back up the path one vertex at a time, forming new paths that are as long as possible until no more edges can be added. Since the graph has a finite number of edges and is connected, this process ends with the production of a spanning tree. Each vertex that ends a path at a stage of the algorithm will be a leaf in the rooted tree, and each vertex where a path is constructed starting at this vertex will be an internal vertex. The reader should note the recursive nature of this procedure. Also, note that if the vertices in the graph are ordered, the choices of edges at each stage of the procedure are all determined when we always choose the first vertex in the ordering that is available. However, we will not always explicitly order the vertices of a graph.

Depth-first search is also called **backtracking,** since the algorithm returns to vertices previously visited to add paths. The following example illustrates backtracking.

EXAMPLE 3 Use a depth-first search to find a spanning tree for the graph G shown in Figure 6.

Solution: The steps used by a depth-first search to produce a spanning tree of G are shown in Figure 7. We arbitrarily start with the vertex f. A path is built by successively adding edges incident with vertices not already in the path, as long as this is possible. This produces a path f, g, h, k, j (note that other paths could have been built). Next,

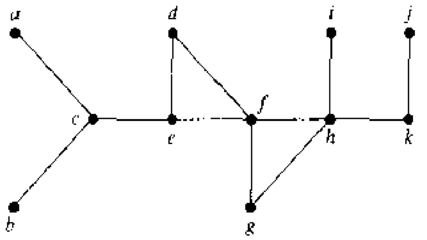


FIGURE 6 The Graph *G*.

backtrack to *k*. There is no path beginning at *k* containing vertices not already visited. So we backtrack to *h*. Form the path *h*, *i*. Then backtrack to *h*, and then to *f*. From *f* build the path *f*, *d*, *e*, *c*, *a*. Then backtrack to *c* and form the path *c*, *b*. This produces the spanning tree. ∎

We can also produce a spanning tree of a simple graph by the use of a **breadth-first search.** Again, a rooted tree will be constructed, and the underlying undirected graph of this rooted tree forms the spanning tree. Arbitrarily choose a root from the vertices of the graph. Then add all edges incident to this vertex. The new vertices added at this stage become the vertices at level 1 in the spanning tree. Arbitrarily order them. Next, for each vertex at level 1, visited in order, add each edge incident to this vertex to the tree as long as it does not produce a simple circuit. Arbitrarily order the children of each vertex at level 1. This produces the vertices at level 2 in the tree. Follow the same procedure until all the vertices in the tree have been added. The procedure ends since there are only a finite number of edges in the graph. A spanning tree is produced since we have produced a tree containing every vertex of the graph. An example of a breadth-first search follows.

EXAMPLE 4 Use a breadth-first search to find a spanning tree for the graph shown in Figure 8.

Solution: The steps of the breadth-first search procedure are shown in Figure 9. We choose the vertex *e* to be the root. Then we add edges incident with all vertices adjacent to *e*, so that edges from *e* to *b*, *d*, *f*, and *i* are added. These four vertices are at level 1 in the tree. Next, add the edges from these vertices at level 1 to adjacent vertices not already in the tree. Hence, the edges from *b* to *a* and *c* are added, as are edges from *d*

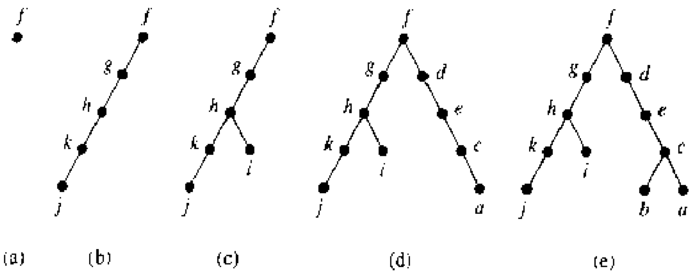


FIGURE 7 A Depth-First Search of *G*.

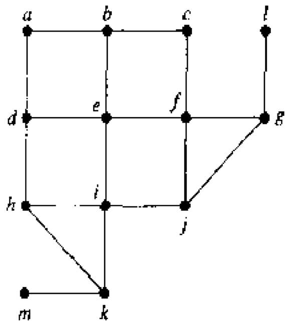


FIGURE 8 A Graph G.

to h, from f to j and g, and from i to k. The new vertices a, c, h, j, g, and k are at level 2. Next, add edges from these vertices to adjacent vertices not already in the graph. This adds edges from g to l and from k to m. ■

BACKTRACKING

There are problems that can be solved only by performing an exhaustive search of all possible solutions. One way to search systematically for a solution is to use a decision tree, where each internal vertex represents a decision and each leaf a possible solution. To find a solution via backtracking, first make a sequence of decisions in an attempt to reach a solution as long as this is possible. The sequence of decisions can be represented by a path in the decision tree. Once it is known that no solution can result from any further sequence of decisions, backtrack to the parent of the current vertex and work toward a solution with another series of decisions, if this is possible. The procedure continues until a solution is found, or it is established that no solution exists. The following examples illustrate the usefulness of backtracking.

EXAMPLE 5 Graph Colorings How can backtracking be used to decide whether a graph can be colored using n colors?

Solution: We can solve this problem using backtracking in the following way. First pick some vertex a and assign it color 1. Then pick a second vertex b, and if b is not adjacent to a, assign it color 1. Otherwise, assign color 2 to b. Then go on to a third

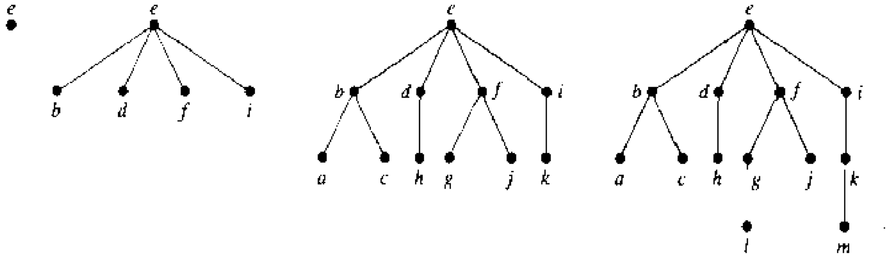


FIGURE 9 A Breadth-First Search of G.

vertex c. Use color 1, if possible, for c. Otherwise use color 2, if this is possible. Only if neither color 1 nor color 2 can be used should color 3 be used. Continue this process as long as it is possible to assign one of the n colors to each additional vertex, always using the first allowable color in the list. If a vertex is reached that cannot be colored by any of the n colors, backtrack to the last assignment made and change the coloring of the last vertex colored, if possible, using the next allowable color in the list. If it is not possible to change this coloring, backtrack further to previous assignments, one step back at a time, until is is possible to change a coloring of a vertex. Then continue assigning colors of additional vertices as long as possible. If a coloring using n colors exists, backtracking will produce it. (Unfortunately this procedure can be extremely inefficient.)

In particular, consider the problem of coloring the graph shown in Figure 10 with three colors. The tree shown in Figure 10 illustrates how backtracking can be used to construct a 3-coloring. In this procedure, red is used first, then blue, and finally green. This simple example can obviously be done without backtracking, but it is a good illustration of the technique.

In this tree, the initial path from the root, which represents the assignment of red to a, leads to a coloring with a red, b blue, c red, and d green. It is impossible to color e using any of the three colors when a, b, c, and d are colored in this way. So, backtrack to the parent of the vertex representing this coloring. Since no other color can be used for d, backtrack one more level. Then change the color of c to green. We obtain a coloring of the graph by then assigning red to d and green to e. ■

EXAMPLE 6 *The n-Queens Problem* The n-queens problem asks how n queens can be placed on an $n \times n$ chessboard so that no two queens can attack one another. How can backtracking be used to solve the n-queens problem?

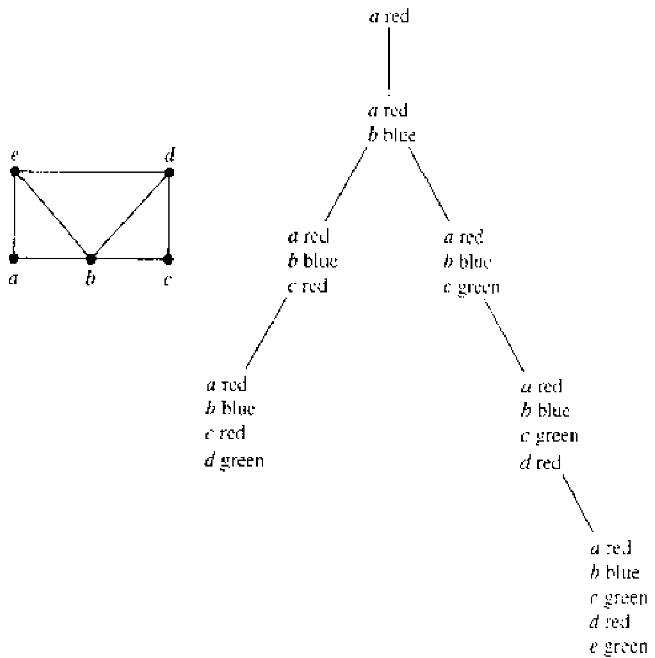

a red
a red b blue
e
d
a
b
c
a red b blue c red
a red b blue c green
a red b blue c red d green
a red b blue c green d red
a red b blue c green d red e green

FIGURE 10 Coloring a Graph Using Backtracking.

Solution: To solve this problem we must find n positions on an $n \times n$ chess board so that no two of these positions are in the same row, same column, or in the same diagonal [a diagonal consists of all positions (i, j) with $i + j = m$ for some m, or $i - j = m$ for some m]. We will use backtracking to solve the n-queens problem. We start with an empty chessboard. At stage $k + 1$ we attempt putting an additional queen on the board in the $(k + 1)$st column, where there are already queens in the first k columns. We examine squares in the $(k + 1)$st column starting with the square in the first row, looking for a position to place this queen so that is not in the same row or on the same diagonal as a queen already on the board. (We already know it is not in the same column.) If it is impossible to find a position to place the queen in the $(k + 1)$st column, backtrack to the placement of the queen in the kth column, and place this queen in the next allowable row in this column, if such a row exists. If no such row exists, backtrack further.

In particular, Figure 11 displays a backtracking solution to the four-queens problem. In this solution, we place a queen in the first row and column. Then we put a queen in the third row of the second column. However, this makes it impossible to place a queen in the third column. So we backtrack and put a queen in the fourth row of the second column. When we do this, we can place a queen in the second row of the third column. But there is no way to add a queen to the fourth column. This shows that no solution results when a queen is placed in the first row and column. We backtrack to the empty chessboard, and place a queen in the second row of the first column. This leads to a solution as shown in Figure 11. ■

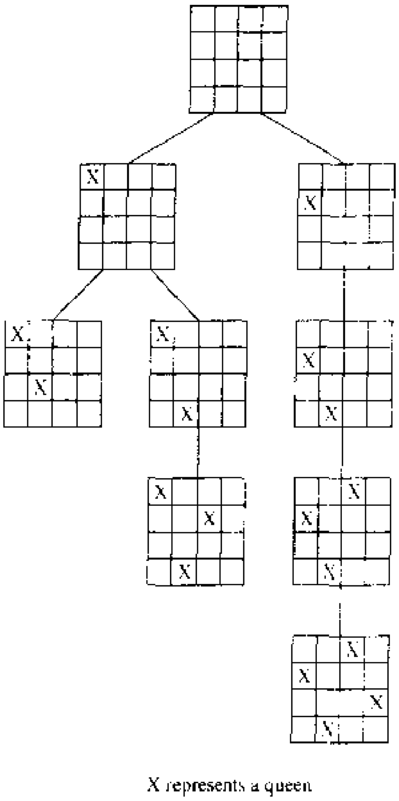


FIGURE 11 A Backtracking Solution of the Four-Queens Problem.

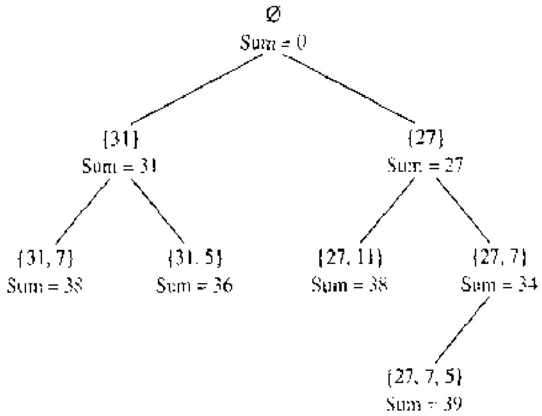

∅
Sum = 0
{31}
Sum = 31
{27}
Sum = 27
{31, 7}
Sum = 38
{31, 5}
Sum = 36
{27, 11}
Sum = 38
{27, 7}
Sum = 34
{27, 7, 5}
Sum = 39

FIGURE 12 **Find a Sum Equal to 39 Using Backtracking.**

EXAMPLE 7 Sums of Subsets Consider the following problem. Given a set of positive integers $x_1, x_2, \ldots, x_n$, find a subset of this set of integers that has M as its sum. How can backtracking be used to solve this problem?

Solution: We start with a sum with no terms. We build up the sum by successively adding terms. An integer in the sequence is included if the sum remains less than M when this integer is added to the sum. If a sum is reached so that the addition of any term is greater than M, backtrack by dropping the last term of the sum.

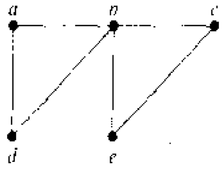
Figure 12 displays a backtracking solution to the problem of finding a subset of $\{31, 27, 15, 11, 7, 5\}$ with sum equal to 39. ■

Exercises

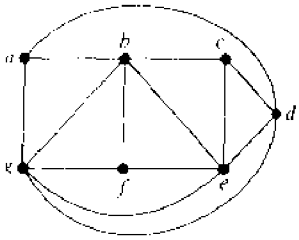
1. How many edges must be removed from a connected graph with n vertices and m edges to produce a spanning tree?

In Exercises 2–6 find a spanning tree for the graph shown by removing edges in simple circuits.

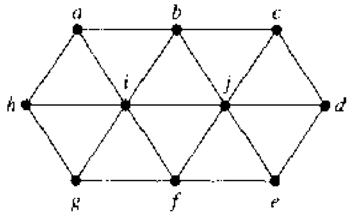
2.



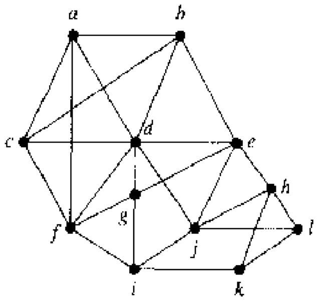
3.



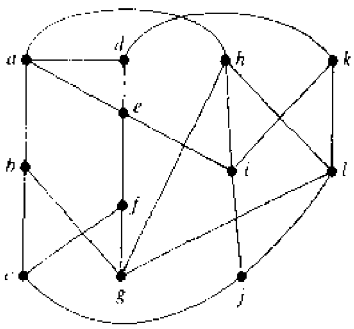
4.


a
b
c
h
i
j
d
g
f
e

5.


a
b
c
d
e
h
l
f
g
j
i
k

6.


a
d
h
k
e
b
i
l
f
c
g
j

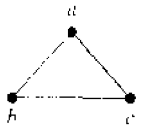
7. Find a spanning tree for each of the following graphs.

a) K_5 **b)** $K_{4,4}$ **c)** $K_{1,6}$

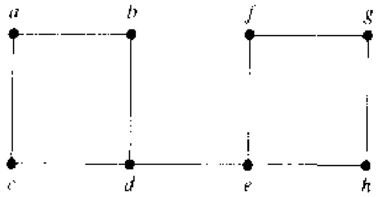
d) Q_3 **e)** C_5 **f)** W_5

In Exercises 8–10 draw *all* the spanning trees of the given simple graphs.

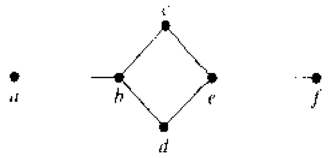
8.



9.



10.



***11.** How many different spanning trees does each of the following simple graphs have?

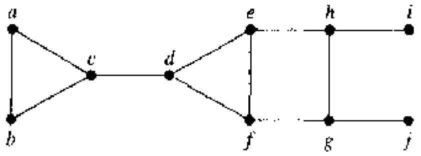
a) K_3 **b)** K_4 **c)** $K_{2,2}$ **d)** C_5

***12.** How many nonisomorphic spanning trees does each of the following simple graphs have?

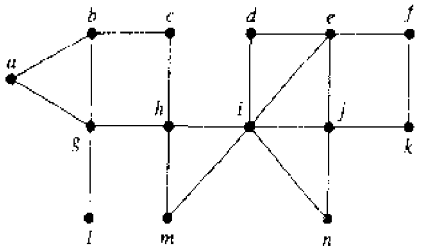
a) K_3 **b)** K_4 **c)** K_5

In Exercises 13–15 use a depth-first search to produce a spanning tree for the given simple graph. Choose a as the root of this spanning tree and assume that the vertices are ordered alphabetically.

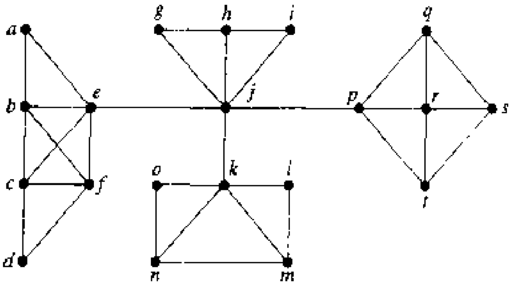
13.


a
e
h
i
c
d
b
f
g
j

14.


b
c
d
e
f
a
h
i
j
g
k
l
m
n

15.

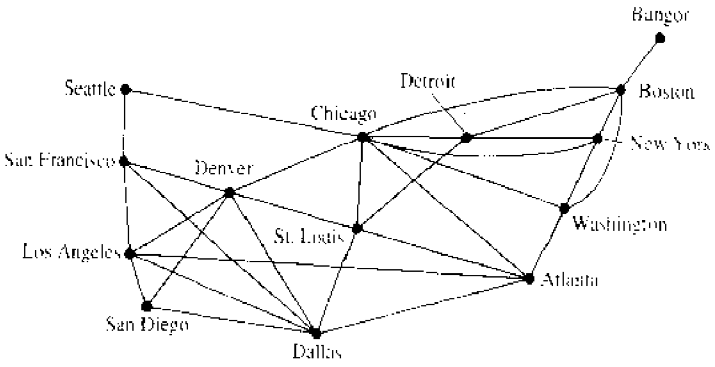

a
g
h
i
q
e
j
p
r
s
b
o
k
l
c
f
t
d
n
m

16. Use a breadth-first search to produce a spanning tree for each of the simple graphs in Exercises 13–15. Choose a as the root of each spanning tree.

17. Suppose that an airline must reduce its flight schedule to save money. If its original routes are as illustrated in the figure at the bottom of the page, which flights can be discontinued to retain service between all pairs of cities (where it may be necessary to combine flights to fly from one city to another)?

18. When must an edge of a connected simple graph be in every spanning tree for this graph?

19. Which connected simple graphs have exactly one spanning tree?


Bangor
Seattle
Detroit
Boston
Chicago
New York
San Francisco
Denver
Washington
St. Louis
Los Angeles
Atlanta
San Diego
Dallas

20. Explain how a breadth-first search or a depth-first search can be used to order the vertices of a connected graph.

***21.** Write the depth-first search procedure in pseudocode.

***22.** Write the breadth-first search procedure in pseudocode.

***23.** Show that the length of the shortest path between vertices v and u in a connected simple graph equals the level number of u in the breadth-first spanning tree of G with root v.

24. Use backtracking to try to find a coloring of each of the graphs in Exercises 5–7 of Section 7.8 using three colors.

25. Use backtracking to solve the n-queens problem for the following values of n.

a) $n = 3$ **b)** $n = 5$ **c)** $n = 6$

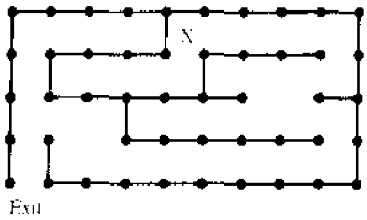
26. Use backtracking to find a subset, if it exists, of the set $\{27, 24, 19, 14, 11, 8\}$ with sum

a) 20. **b)** 41. **c)** 60.

27. Explain how backtracking can be used to find a Hamilton path or circuit in a graph.

28. **a)** Explain how backtracking can be used to find the way out of a maze, given a starting position and the exit position. Consider the maze divided into positions, where at each position the set of available moves includes one to four possibilities (up, down, right, left).

b) Find a path from the starting position marked by X to the exit in the following maze.



A **spanning forest** of graph G is a forest that contains every vertex of G such that two vertices are in the same tree of the forest when there is a path in G between these two vertices.

29. Show that every finite simple graph has a spanning forest.

30. How many trees are in the spanning forest of a graph?

31. How many edges must be removed to produce the spanning forest of a graph with n vertices, m edges, and c connected components?

32. Devise an algorithm for constructing the spanning forest of a graph based on deleting edges that form simple circuits.

33. Devise an algorithm for constructing the spanning forest of a graph based on depth-first searching.

34. Devise an algorithm for constructing the spanning forest of a graph based on breadth-first searching.

Let T_1 and T_2 be spanning trees of a graph. The **distance** between T_1 and T_2 is the number of edges in T_1 and T_2 that are not common to T_1 and T_2.

35. Find the distance between each pair of spanning trees shown in Figures 3(c) and 4 of the graph G shown in Figure 2.

***36.** Suppose that T_1, T_2, and T_3 are spanning trees of the simple graph G. Show that the distance between T_1 and T_3 does not exceed the sum of the distance between T_1 and T_2 and the distance between T_2 and T_3.

****37.** Suppose that T_1 and T_2 are spanning trees of a simple graph G. Moreover, suppose that e_1 is an edge in T_1 that is not in T_2. Show that there is an edge e_2 in T_2, that is not in T_1 such that T_1 remains a spanning tree if e_1 is removed from it and e_2 is added to it, and T_2 remains a spanning tree if e_2 is removed from it and e_1 is added to it.

***38.** Show that it is possible to find a sequence of spanning trees leading from any spanning tree to any other by successively removing one edge and adding another.

A **rooted spanning tree** of a directed graph is a rooted tree containing edges of the graph such that every vertex of the graph is an endpoint of one of the edges in the tree.

39. For each of the directed graphs in Exercises 24–28 of Section 7.5 either find a rooted spanning tree of the graph or determine that no such tree exists.

***40.** Show that a connected directed graph in which each vertex has the same in-degree and out-degree has a rooted spanning tree. (*Hint:* Use an Euler circuit.)

***41.** Give an algorithm to build a rooted spanning tree for connected directed graphs in which each vertex has the same in-degree and out-degree.

8.6

Minimum Spanning Trees

INTRODUCTION

web A company plans to build a communications network connecting its five computer centers. Any pair of these centers can be linked with a leased telephone line. Which links

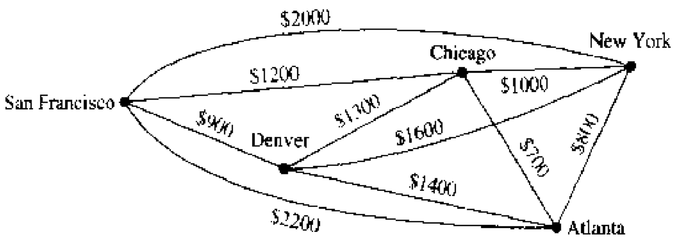


FIGURE 1 A Weighted Graph Showing Monthly Lease Costs for Lines in a Computer Network.

should be made to ensure that there is a path between any two computer centers so that the total cost of the network is minimized? We can model this problem using the weighted graph shown in Figure 1, where vertices represent computer centers, edges represent possible leased lines, and the weights on edges are the monthly lease rates of the lines represented by the edges. We can solve this problem by finding a spanning tree so that the sum of the weights of the edges of the tree is minimized. Such a spanning tree is called a **minimum spanning tree.**

ALGORITHMS FOR MINIMUM SPANNING TREES

A wide variety of problems are solved by finding a spanning tree in a weighted graph such that the sum of the weights of the edges in the tree is a minimum.

DEFINITION 1. A *minimum spanning tree* in a connected weighted graph is a spanning tree that has the smallest possible sum of weights of its edges.

We will present two algorithms for constructing minimum spanning trees. Both proceed by successively adding edges of smallest weight from those edges with a specified property that have not already been used. These algorithms are examples of **greedy algorithms.** A greedy algorithm is a procedure that makes an optimal choice at each of its steps. Optimizing at each stage of an algorithm does not guarantee that the optimal overall solution is produced. However, the two algorithms presented in this section for constructing minimum spanning trees are greedy algorithms that do produce optimal solutions.

The first algorithm that we will discuss was given by Robert Prim in 1957, although the basic ideas of this algorithm have an earlier origin. To carry out **Prim's algorithm,** begin by choosing any edge with smallest weight, putting it into the spanning tree. Successively add to the tree edges of minimum weight that are incident to a vertex already in the tree and not forming a simple circuit with those edges already in the tree. Stop when $n - 1$ edges have been added.

web

Robert Clay Prim (born 1921). Robert Prim, born in Sweetwater, Texas, received his B.S. in Electrical Engineering in 1941 and his Ph.D. in Mathematics from Princeton University in 1949. He was an engineer at the General Electric Company from 1941 until 1944, an engineer and mathematician at the United States Naval Ordnance Lab from 1944 until 1949, and a research associate at Princeton University from 1948 until 1949. Among the other positions he has held are director of mathematics and mechanics research at Bell Telephone Laboratories from 1958 until 1961 and vice president of research at Sandia Corporation. He is currently retired.

Later in this section, we will prove that this algorithm produces a minimum spanning tree for any connected weighted graph. Algorithm 1 gives a pseudocode description of Prim's algorithm.

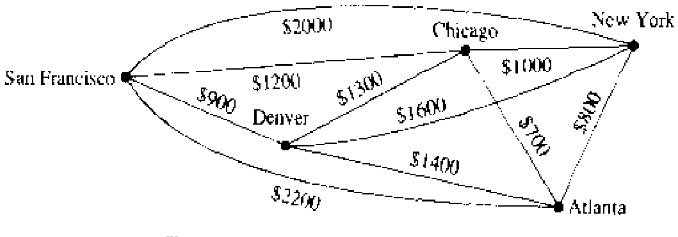
ALGORITHM 1 Prim's Algorithm.

```
procedure Prim(G: weighted connected undirected graph with $n$
vertices)
T := a minimum-weight edge
for $i$ := 1 to $n - 2$
begin
  $e$ := an edge of minimum weight incident to a vertex in $T$ and not
     forming a simple circuit in $T$ if added to $T$
  $T$ := $T$ with $e$ added
end {$T$ is a minimum spanning tree of $G$}
```

Note that the choice of an edge to add at a stage of the algorithm is not determined when there is more than one edge with the same weight that satisfies the appropriate criteria. We need to order the edges to make the choices deterministic. We will not worry about this in the remainder of the section. Also note that there may be more than one minimum spanning tree for a given connected weighted simple graph. (See Exercise 9.) The following examples illustrate how Prim's algorithm is used.

EXAMPLE 1 Use Prim's algorithm to design a minimum-cost communications network connecting all the computers represented by the graph in Figure 1.

Solution: We solve this problem by finding a minimum spanning tree in the graph in Figure 1. Prim's algorithm is carried out by choosing an initial edge of minimum weight and successively adding edges of minimum weight that are incident to a vertex in the tree and that do not form simple circuits. The edges in color in Figure 2 show a minimum spanning tree produced by Prim's algorithm, with the choice made at each step displayed. ■



Choice	Edge	Cost
1	{ Chicago, Atlanta }	$ 700
2	{ Atlanta, New York }	$ 800
3	{ Chicago, San Francisco }	$1200
4	{ San Francisco, Denver }	$ 900
	Total:	$3600

FIGURE 2 A Minimum Spanning Tree for the Weighted Graph in Figure 1.

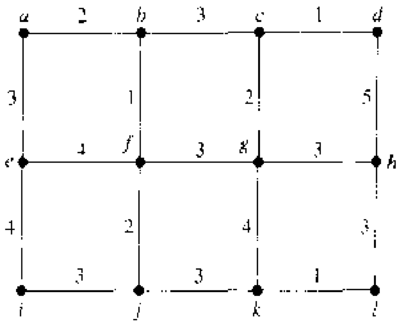
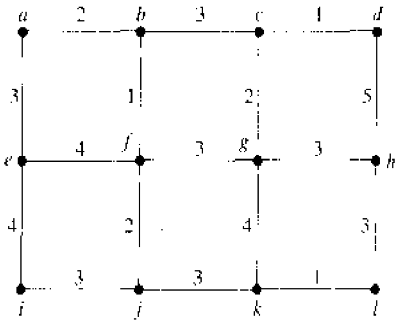


FIGURE 3 A Weighted Graph.

EXAMPLE 2 Use Prim's algorithm to find a minimum spanning tree in the graph shown in Figure 3.

Solution: A minimum spanning tree constructed using Prim's algorithm is shown in Figure 4. The successive edges chosen are displayed. ■

The second algorithm we will discuss was discovered by Joseph Kruskal in 1956, although the basic ideas it uses were described much earlier. To carry out **Kruskal's algorithm,** choose an edge in the graph with minimum weight.



(a)

Choice	Edge	Weight
1	$\{b, f\}$	1
2	$\{a, b\}$	2
3	$\{f, j\}$	2
4	$\{a, e\}$	3
5	$\{i, j\}$	3
6	$\{f, g\}$	3
7	$\{c, g\}$	2
8	$\{c, d\}$	1
9	$\{g, h\}$	3
10	$\{h, l\}$	3
11	$\{k, l\}$	1
	Total:	24

(b)

FIGURE 4 A Minimum Spanning Tree Produced Using Prim's Algorithm.

Joseph Bernard Kruskal (born 1928). Joseph Kruskal, born in New York City, attended the University of Chicago and received his Ph.D. from Princeton University in 1954. He was an instructor in mathematics at Princeton and at the University of Wisconsin, and later he was an assistant professor at the University of Michigan. In 1959 he became a member of the technical staff at Bell Laboratories, a position he continues to hold. His current research interests include statistical linguistics and psychometrics. Besides his work on minimum spanning trees, Kruskal is also known for contributions to multidimensional scaling. Kruskal discovered his algorithm for producing minimum spanning trees when he was a second-year graduate student. He was not sure his 2½-page paper on this subject was worthy of publication but was convinced by others to submit it.

Historical Note: Joseph Kruskal and Robert Prim developed their algorithms for constructing minimum spanning trees in the mid-1950s. However, they were not the first people to discover such algorithms. For example, the work of the anthropologist Jan Czekanowski, in 1909, contains many of the ideas required to find minimum spanning trees. In 1926, Otakar Boruvka described methods for constructing minimum spanning trees in work relating to the construction of electric power networks.

Successively add edges with minimum weight that do not form a simple circuit with those edges already chosen. Stop after $n - 1$ edges have been selected.

The proof that Kruskal's algorithm produces a minimum spanning tree for every connected weighted graph is left as an exercise at the end of this section. Pseudocode for Kruskal's algorithm is given in Algorithm 2.

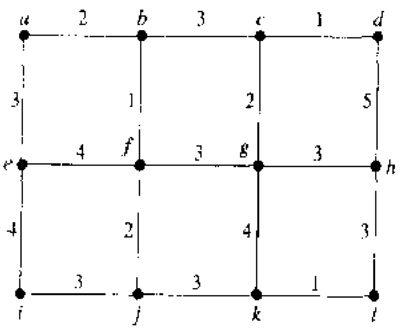
ALGORITHM 2 Kruskal's Algorithm.

```
procedure Kruskal(G: weighted connected undirected graph with $n$
  vertices)
T := empty graph
for $i$ := 1 to $n - 1$
begin
  $e$ := any edge in $G$ with smallest weight that does not form a
    simple circuit when added to $T$
  $T$ := $T$ with $e$ added
end {$T$ is a minimum spanning tree of $G$}
```

The reader should note the difference between Prim's and Kruskal's algorithms. In Prim's algorithm edges of minimum weight that are incident to a vertex already in the tree, and not forming a circuit, are chosen; whereas, in Kruskal's algorithm edges of minimum weight that are not necessarily incident to a vertex already in the tree, and that do not form a circuit, are chosen. Note that as in Prim's algorithm, if the edges are not ordered, there may be more than one choice for the edge to add at a stage of this procedure. Consequently, the edges need to be ordered for the procedure to be deterministic. The following example illustrates how Kruskal's algorithm is used.

EXAMPLE 3 Use Kruskal's algorithm to find a minimum spanning tree in the weighted graph shown in Figure 3.

Solution: A minimum spanning tree and the choices of edges at each stage of Kruskal's algorithm are shown in Figure 5. ∎



(a)

Choice	Edge	Weight
1	$\{c, d\}$	1
2	$\{k, l\}$	1
3	$\{b, f\}$	1
4	$\{c, g\}$	2
5	$\{a, b\}$	2
6	$\{f, j\}$	2
7	$\{b, c\}$	3
8	$\{j, k\}$	3
9	$\{g, h\}$	3
10	$\{i, j\}$	3
11	$\{a, e\}$	3
	Total:	24

(b)

FIGURE 5 A Minimum Spanning Tree Produced by Kruskal's Algorithm.

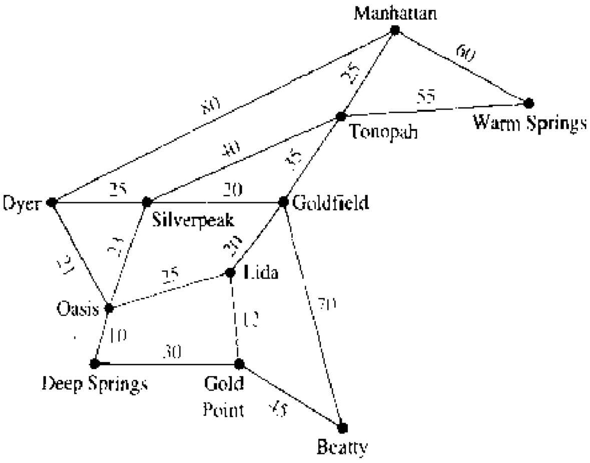
We will now prove that Prim's algorithm produces a minimum spanning tree of a connected weighted graph.

Proof: Let G be a connected weighted graph. Suppose that the successive edges chosen by Prim's algorithm are $e_1, e_2, \ldots, e_{n-1}$. Let S be the tree with $e_1, e_2, \ldots, e_{n-1}$ as its edges, and let S_k be the tree with $e_1, e_2, \ldots, e_k$ as its edges. Let T be a minimum spanning tree of G containing the edges $e_1, e_2, \ldots, e_k$, where k is the maximum integer with the property that a minimum spanning tree exists containing the first k edges chosen by Prim's algorithm. The theorem follows if we can show that $S = T$.

Suppose that $S \neq T$, so that $k < n - 1$. Consequently, T contains $e_1, e_2, \ldots, e_k$, but not e_{k+1}. Consider the graph made up of T together with e_{k+1}. Since this graph is connected and has n edges, too many edges to be a tree, it must contain a simple circuit. This simple circuit must contain e_{k+1} since there was no simple circuit in T. Furthermore, there must be an edge in the simple circuit that does not belong to S_{k+1} since S_{k+1} is a tree. By starting at an endpoint of e_{k+1} that is also an endpoint of one of the edges $e_1, \ldots, e_k$, and following the circuit until it reaches an edge not in S_{k+1}, we can find an edge e not in S_{k+1} that has an endpoint that is also an endpoint of one of the edges $e_1, e_2, \ldots, e_k$. By deleting e from T and adding e_{k+1}, we obtain a tree T' with $n - 1$ edges (it is a tree since it has no simple circuits). Note that the tree T' contains $e_1, e_2, \ldots, e_k, e_{k+1}$. Furthermore, since e_{k+1} was chosen by Prim's algorithm at the $(k + 1)$st step, and e was also available at that step, the weight of e_{k+1} is less than or equal to the weight of e. From this observation it follows that T' is also a minimum spanning tree, since the sum of the weights of its edges does not exceed the sum of the weights of the edges of T. This contradicts the choice of k as the maximum integer so that a minimum spanning tree exists containing $e_1, \ldots, e_k$. Hence, $k = n - 1$, and $S = T$. It follows that Prim's algorithm produces a minimum spanning tree. ☐

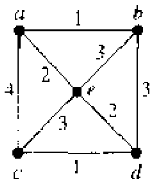
Exercises

1. The roads represented by the following graph are all unpaved. The lengths of the roads between pairs of towns are represented by edge weights. Which roads should be paved so that there is a path of paved roads between each pair of towns so that a minimum road length is paved? (*Note:* These towns are in Nevada.)

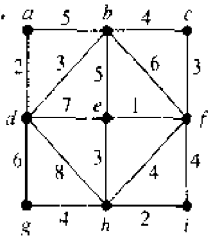


In Exercises 2–4 use Prim's algorithm to find a minimum spanning tree for the given weighted graph.

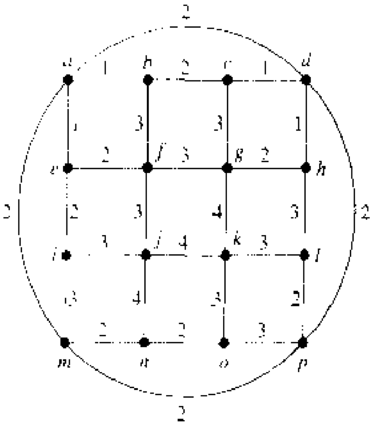
2.



3.



4.



5. Use Kruskal's algorithm to design the communications network described at the beginning of the section.

6. Use Kruskal's algorithm to find a minimum spanning tree for the weighted graph in Exercise 2.

7. Use Kruskal's algorithm to find a minimum spanning tree for the weighted graph in Exercise 3.

8. Use Kruskal's algorithm to find a minimum spanning tree for the weighted graph in Exercise 4.

9. Find a connected weighted simple graph with the fewest edges possible that has more than one minimum spanning tree.

10. A **minimum spanning forest** in a weighted graph is a spanning forest with minimal weight. Explain how Prim's and Kruskal's algorithms can be adapted to construct minimum spanning forests.

A **maximum spanning tree** of a connected weighted undirected graph is a spanning tree with the largest possible weight.

11. Devise an algorithm similar to Prim's algorithm for constructing a maximum spanning tree of a connected weighted graph.

12. Devise an algorithm similar to Kruskal's algorithm for constructing a maximum spanning tree of a connected weighted graph.

13. Find a maximum spanning tree for the weighted graph in Exercise 2.

14. Find a maximum spanning tree for the weighted graph in Exercise 3.

15. Find a maximum spanning tree for the weighted graph in Exercise 4.

16. Find the second least expensive communications network connecting the five computer centers in the problem posed at the beginning of the section.

*17. Devise an algorithm for finding the second shortest spanning tree in a connected weighted graph.

*18. Show that an edge with smallest weight in a connected weighted graph must be part of any minimum spanning tree.

19. Show that there is a unique minimum spanning tree in a connected weighted graph if the weights of the edges are all different.

20. Suppose that the computer network connecting the cities in Figure 1 must contain a direct link between New York and Denver. What other links should be included so that there is a link between every two computer centers and the cost is minimized?

21. Find a spanning tree with minimal total weight containing the edges $\{e, i\}$ and $\{g, k\}$ in the weighted graph in Figure 3.

22. Describe an algorithm for finding a spanning tree with minimal weight containing a specified set of edges in a connected weighted undirected simple graph.

23. Express the algorithm devised in Exercise 22 in pseudocode.

Sollin's algorithm produces a minimum spanning tree from a connected weighted simple graph $G = (V, E)$ by successively adding groups of edges. Suppose that the vertices in V are ordered. This produces an ordering of the edges where $\{u_0, v_0\}$ precedes $\{u_1, v_1\}$ if u_0 precedes u_1 or if $u_0 = u_1$ and v_0 precedes v_1. The algorithm begins by simultaneously choosing the edge of least weight incident to each vertex. The first edge in the ordering is taken in the case of ties. This produces a graph with no simple circuits, that is, a forest of trees (Exercise 24 asks for a proof of this fact). Next, simultaneously choose for each tree in the forest the shortest edge between a vertex in this tree and a vertex in a different tree. Again the first edge in the ordering is chosen in the case of ties. (This produces a graph with no simple circuits containing fewer trees than were present before this step; see Exercise 24.) Continue the process of simultaneously adding edges connecting trees until $n - 1$ edges have been chosen. At this stage a minimum spanning tree has been constructed.

*24. Show that the addition of edges at each stage of Sollin's algorithm produces a forest.

25. Use Sollin's algorithm to produce a minimum spanning tree for the weighted graph shown in
    a) Figure 1. b) Figure 3.

*26. Express Sollin's algorithm in pseudocode.

**27. Prove that Sollin's algorithm produces a minimum spanning tree in a connected undirected weighted graph.

*28. Show that the first step of Sollin's algorithm produces a forest containing at least $\lceil n/2 \rceil$ edges.

*29. Show that if there are r trees in the forest at some intermediate step of Sollin's algorithm, then at least $\lceil r/2 \rceil$ edges are added by the next iteration of the algorithm.

*30. Show that no more than $\lfloor n/2^k \rfloor$ trees remain after the first step of Sollin's algorithm has been carried out and the second step of the algorithm has been carried out $k - 1$ times.

*31. Show that Sollin's algorithm requires at most $\log n$ iterations to produce a minimum spanning tree from a connected undirected weighted graph with n vertices.

32. Prove that Kruskal's algorithm produces minimum spanning trees.

Key Terms and Results

TERMS

tree: a connected undirected graph with no simple circuits

forest: an undirected graph with no simple circuits

rooted tree: a directed graph with a specified vertex, called the root, such that there is a unique path to any other vertex from this root

subtree: a subgraph of a tree that is also a tree

parent of v in a rooted tree: the vertex u such that (u, v) is an edge of the rooted tree

child of a vertex v in a rooted tree: any vertex with v as its parent

sibling of a vertex v in a rooted tree: a vertex with the same parent as v

ancestor of a vertex v in a rooted tree: any vertex on the path from the root to v

descendant of a vertex v in a rooted tree: any vertex that has v as an ancestor

internal vertex: a vertex that has children

leaf: a vertex with no children

level of a vertex: the length of the path from the root to this vertex

height of a tree: the largest level of the vertices of a tree

***m*-ary tree:** a tree with the property that every internal vertex has no more than m children

full *m*-ary tree: a tree with the property that every internal vertex has exactly m children

binary tree: an m-ary tree with $m = 2$ (each child may be designated as a left or a right child of its parent)

ordered tree: a tree in which the children of each internal vertex are linearly ordered

balanced tree: a tree in which every vertex is at level h or $h - 1$, where h is the height of the tree

binary search tree: a binary tree in which the vertices are labeled with items so that a label of a vertex is greater than the labels of all vertices in the left subtree of this vertex and is less than the labels of all vertices in the right subtree of this vertex

decision tree: a rooted tree where each vertex represents a possible outcome of a decision and the leaves represent the possible solutions

prefix code: a code that has the property that the code of a character is never a prefix of the code of another character

tree traversal: a listing of the vertices of a tree

preorder traversal: a listing of the vertices of an ordered rooted tree defined recursively by specifying that the root is listed, followed by the first subtree, followed by the other subtrees in the order they occur from left to right

inorder traversal: a listing of the vertices of an ordered rooted tree defined recursively by specifying that the first subtree is listed, followed by the root, followed by the other subtrees in the order they occur from left to right

postorder traversal: a listing of the vertices of an ordered rooted tree defined recursively by specifying that the subtrees are listed in the order they occur from left to right, followed by the root

infix notation: the form of an expression (including a full set of parentheses) obtained from an inorder traversal of the binary tree representing this expression

prefix (or Polish) notation: the form of an expression obtained from a preorder traversal of the tree representing this expression

postfix (or reverse Polish) notation: the form of an expression obtained from a postorder traversal of the tree representing this expression

sorting problem: a problem in which a list of items is to be put into increasing order

spanning tree: a tree containing all vertices of a graph

minimum spanning tree: a spanning tree with smallest possible sum of weights of its edges

greedy algorithm: an algorithm that optimizes by making the optimal choice at each step

RESULTS

A graph is a tree if and only if there is a unique simple path between any of its vertices.

A tree with n vertices has $n - 1$ edges.

A full m-ary tree with i internal vertices has $mi + 1$ vertices.

The relationships between the numbers of vertices, leaves, and internal vertices in a full m-ary tree (see Theorem 4 in Section 8.1).

There are at most m^h leaves in an m-ary tree of height h.

If an m-ary tree has l leaves, its height h is at least $\lceil \log_m l \rceil$. If the tree is also full and balanced, then its height is $\lceil \log_m l \rceil$.

the bubble sort: a sorting procedure that is carried out using passes where successive items that are out of order are interchanged

the merge sort: a sorting procedure that is carried out by successively merging pairs of sublists of the original list

depth-first search, or backtracking: a procedure for constructing a spanning tree by adding edges that form a path until this is not possible, and then moving back up the path until a vertex is found where a new path can be formed

breadth-first search: a procedure for constructing a spanning tree that successively adds all edges incident to the last set of edges added, unless a simple circuit is formed

Prim's algorithm: a procedure for producing a minimum spanning tree in a weighted graph that successively adds edges with minimal weight among all edges incident to a vertex already in the tree such that no edge produces a simple circuit when it is added

Kruskal's algorithm: a procedure for producing a minimum spanning tree in a weighted graph that successively adds edges of least weight that are not already in the tree such that no edge produces a simple circuit when it is added

Review Questions

1. a) Define a tree. b) Define a forest.
2. Can there be two different simple paths between the vertices of a tree?
3. Give at least three examples of how trees are used in modeling.
4. a) Define a rooted tree and the root of such a tree.
   b) Define the parent of a vertex and a child of a vertex in a rooted tree.
   c) What are an internal vertex, a leaf, and a subtree in a rooted tree?
   d) Draw a rooted tree with at least 10 vertices, where the degree of each vertex does not exceed 3. Identify the root, the parent of each vertex, the children of each vertex, the internal vertices, and the leaves.
5. a) How many edges does a tree with n vertices have?
   b) What do you need to know to determine the number of edges in a forest with n vertices?
6. a) Define a full m-ary tree.
   b) How many vertices does a full m-ary tree have if it has i internal vertices? How many leaves does the tree have?
7. a) What is the height of a rooted tree?
   b) What is a balanced tree?
   c) How many leaves can an m-ary tree of height h have?
8. a) What is a binary search tree?
   b) Describe an algorithm for constructing a binary search tree.
   c) Form a binary search tree for the words *vireo, warbler, egret, grosbeak, nuthatch,* and *kingfisher.*
9. a) What is a prefix code?
   b) How can a prefix code be represented by a binary tree?
10. a) Define preorder, inorder, and postorder tree traversal.
    b) Give an example of preorder, postorder, and inorder traversal of a binary tree of your choice with at least 12 vertices.
11. a) Explain how to use preorder, inorder, and postorder traversals to find the prefix, infix, and postfix forms of an arithmetic expression.
    b) Draw the ordered rooted tree that represents $((x - 3) * ((x/4) + (x - y) \uparrow 3))$.
    c) Find the prefix and postfix forms of the expression in part (b).
12. Show that the number of comparisons used by a sorting algorithm is at least $\lceil \log n! \rceil$.
13. a) Describe the bubble sort algorithm.
    b) Use the bubble sort algorithm to put the list 5, 2, 4, 1, 3 in increasing order.
    c) Give a big-O estimate for the number of comparisons used by the bubble sort.
14. a) Describe the merge sort algorithm.
    b) Use the merge sort algorithm to put the list 4, 10, 1, 5, 3, 8, 7, 2, 6, 9 in increasing order.
    c) Give a big-O estimate for the number of comparisons used by the merge sort.
15. a) What is a spanning tree of a simple graph?
    b) Which simple graphs have spanning trees?
    c) Describe at least two different applications that require that a spanning tree of a simple graph be found.
16. a) Describe two different algorithms for finding a spanning tree in a simple graph.
    b) Illustrate how the two algorithms you described in (a) can be used to find the spanning tree of a simple graph, using a graph of your choice with at least eight vertices and 15 edges.
17. a) Explain how backtracking can be used to determine whether a simple graph can be colored using n colors.
    b) Show, with an example, how backtracking can be used to show that a graph with a chromatic number equal to 4 cannot be colored with three colors, but can be colored with four colors.
18. a) What is a minimum spanning tree of a connected weighted graph?
    b) Describe at least two different applications that require that a minimum spanning tree of a connected weighted graph be found.
19. a) Describe Kruskal's algorithm and Prim's algorithm for finding minimum spanning trees.
    b) Illustrate how Kruskal's algorithm and Prim's algorithm are used to find a minimum spanning tree, using a weighted graph with at least eight vertices and 15 edges.

Supplementary Exercises

***1.** Show that a simple graph is a tree if and only if it contains no simple circuits and the addition of an edge connecting two nonadjacent vertices produces a new graph that has exactly one simple circuit (where circuits that contain the same edges are not considered different).

***2.** How many nonisomorphic rooted trees are there with six vertices?

3. Show that every tree with at least one edge must have at least two pendant vertices.

4. Show that a tree with n vertices which has $n - 1$ pendant vertices must be isomorphic to $K_{1,n-1}$.

5. What is the sum of the degrees of the vertices of a tree with n vertices?

***6.** Suppose that $d_1, d_2, \ldots, d_n$ are n positive integers with sum $2n - 2$. Show that there is a tree which has n vertices so that the degrees of these vertices are $d_1, d_2, \ldots, d_n$.

7. Show that every tree is a planar graph.

8. Show that every tree is bipartite.

9. Show that every forest can be colored using two colors.

A **B-tree of degree** k is a rooted tree such that all its leaves are at the same level, its root has at least two and at most k children unless it is a leaf, and every internal vertex other than the root has at least $\lceil k/2 \rceil$, but no more than k, children. Computer files can be accessed efficiently when B-trees are used to represent them.

10. Draw three different B-trees of degree 3 with height 4.

***11.** Give an upper bound and a lower bound for the number of leaves in a B-tree of degree k with height h.

***12.** Give an upper bound and a lower bound for the height of a B-tree of degree k with n leaves.

A rooted tree T is called an S_k**-tree** if it satisfies the following recursive definition. It is an S_0-tree if it has one vertex. For $k > 0$, T is an S_k-tree if it can be built from two S_{k-1}-trees by making the root of one the root of the S_k-tree and making the root of the other the child of the root of the first S_{k-1}-tree.

13. Draw an S_k-tree for $k = 0, 1, 2, 3, 4$.

14. Show that an S_k-tree has 2^k vertices and a unique vertex at level k. This vertex at level k is called the **handle.**

***15.** Suppose that T is an S_k-tree with handle v. Show that T can be obtained from disjoint trees $T_0, T_1, \ldots, T_{k-1}$, where v is not in any of these trees, where T_i is an S_i-tree for $i = 0, 1, \ldots, k - 1$, by connecting v to r_0 and r_i to r_{i+1} for $i = 0, 1, \ldots, k - 2$.

The listing of the vertices of an ordered rooted tree in **level order** begins with the root, followed by the vertices at level 1 from left to right, followed by the vertices at level 2 from left to right, and so on.

16. List the vertices of the ordered rooted trees in Figures 3 and 9 of Section 8.3 in level order.

17. Devise an algorithm for listing the vertices of an ordered rooted tree in level order.

***18.** Devise an algorithm for determining if a set of universal addresses can be the addresses of the leaves of a rooted tree.

19. Devise an algorithm for constructing a rooted tree from the universal addresses of its leaves.

The **insertion sort** operates by considering the elements in a list one at a time, beginning with the second element. Each element is compared to the previous elements in the list, which have been put in the correct order, and this element is put in the correct position among these, moving the element that was in this position, and all elements to the right of this, one position to the right.

20. Sort the list 3, 2, 4, 5, 1 using an insertion sort.

21. Write the insertion sort in pseudocode.

22. Determine the worst-case complexity of the insertion sort in terms of the number of comparisons used.

23. Suppose that e is an edge in a simple graph that is incident to a pendant vertex. Show that e must be in any spanning tree.

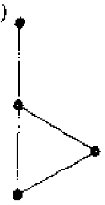
A **cut set** of a graph is a set of edges such that the removal of these edges produces a subgraph with more connected components than in the original graph, but no proper subset of this set of edges has this property.

24. Show that a cut set of a graph must have at least one edge in common with any spanning tree of this graph.

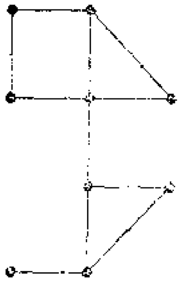
A **cactus** is a connected graph in which no edge is in more than one simple circuit not passing through any vertex other than its initial vertex more than once or its initial vertex other than at its terminal vertex (where two circuits that contain the same edges are not considered different).

25. Which of the following graphs are cacti?

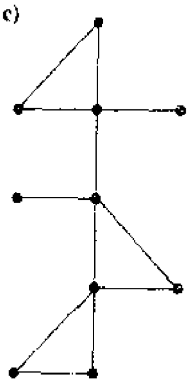
a)



b)



c)



26. Is a tree necessarily a cactus?

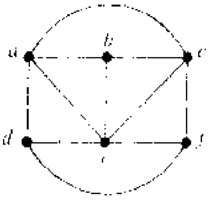
27. Show that a cactus is formed if we add a circuit containing new edges beginning and ending at a vertex of a tree.

***28.** Show that if every circuit not passing through any vertex other than its initial vertex more than once in a connected graph contains an odd number of edges, then this graph must be a cactus.

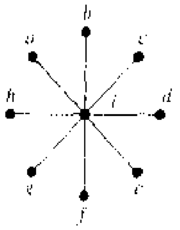
A **degree-constrained spanning tree** of a simple graph G is a spanning tree with the property that the degree of a vertex in this tree cannot exceed some specified bound. Degree-constrained spanning trees are useful in models of transportation systems where the number of roads at an intersection is limited, models of communications networks where the number of links entering a node is limited, and so on.

In Exercises 29–31 find a degree-constrained spanning tree of the given graph where each vertex has degree less than or equal to 3, or show that such a spanning tree does not exist.

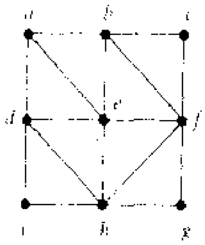
29.



30.



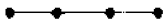
31.



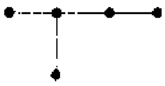
32. Show that a degree-constrained spanning tree of a simple graph in which each vertex has degree not exceeding 2 consists of a single Hamilton path in the graph.

33. A tree with n vertices is called **graceful** if its vertices can be labeled with the integers $1, 2, \ldots, n$ such that the absolute value of the difference of the labels of adjacent vertices are all different. Show that the following trees are graceful.

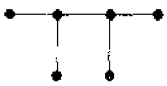
a)



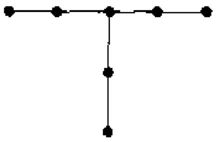
b)



c)



d)



A **caterpillar** is a tree that contains a simple path such that every vertex not contained in this path is adjacent to a vertex in the path.

34. Which of the graphs in Exercise 33 are caterpillars?

35. How many nonisomorphic caterpillars are there with six vertices?

***36.** **a)** Prove or disprove that all trees whose edges form a single path are graceful.

****b)** Prove or disprove that all caterpillars are graceful.

37. Suppose that the first four moves of a tic-tac-toe game are as shown. Explain how a tree can be used to show the possible successive moves of this game. If the player using X goes first, does this player have a strategy that will always win?

a)

X	O	X
	O	

b)

X	O	X
	O	

c)

	O	X
	X	
O		

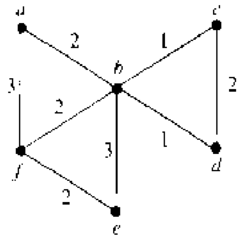
38. Three couples arrive at the bank of a river. Each of the wives is jealous and does not trust her husband when he is with one of the other wives (and perhaps with other people), but not with her. How can six people cross to the other side of the river using a boat that can hold no more than two people so that no husband is alone with a woman other than his wife? Use a graph theory model.

***39.** Suppose that e is an edge in a weighted graph that is incident to a vertex v so that the weight of e does not exceed the weight of any other edge incident to v. Show that there exists a minimum spanning tree containing this edge.

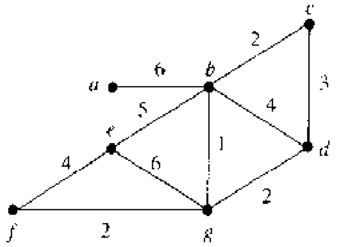
***40.** Show that if no two edges in a weighted graph have the same weight, then the edge with least weight incident to a vertex v is included in every minimum spanning tree.

41. Find a minimum spanning tree of each of the following graphs where the degree of each vertex in the spanning tree does not exceed 2.

a)



b)



Computer Projects

WRITE PROGRAMS WITH THE FOLLOWING INPUT AND OUTPUT.

1. Given the adjacency matrix of an undirected simple graph, determine whether the graph is a tree.

2. Given the adjacency matrix of a rooted tree and a vertex in the tree, find the parent, children, ancestors, descendants, and level of this vertex.

3. Given the list of edges of a rooted tree and a vertex in the tree, find the parent, children, ancestors, descendants, and level of this vertex.

4. Given a list of items, construct a binary search tree containing these items.

5. Given a binary search tree and an item, locate or add this item to the binary search tree.

6. Given the ordered list of edges of an ordered rooted tree, find the universal addresses of its vertices.

7. Given the ordered list of edges of an ordered rooted tree, list its vertices in preorder, inorder, and postorder.

8. Given an arithmetic expression in prefix form, find its value.

9. Given an arithmetic expression in postfix form, find its value.

10. Given a set of n integers, sort them using a bubble sort.

11. Given two sorted lists of integers, merge them into one sorted list, keeping track of the number of comparisons used.

12. Given a set of n integers, sort them using a merge sort.

13. Given the adjacency matrix of a connected undirected simple graph, find a spanning tree for this graph using a depth-first search.

14. Given the adjacency matrix of a connected undirected simple graph, find a spanning tree for this graph using a breadth-first search.

15. Given a set of positive integers and a positive integer N, use backtracking to find a subset of these integers that have N as their sum.

***16.** Given the adjacency matrix of an undirected simple graph, use backtracking to color the graph with three colors, if this is possible.

***17.** Given a positive integer n, solve the n-queens problem using backtracking.

18. Given the list of edges and their weights of a weighted undirected connected graph, use Prim's algorithm to find a minimum spanning tree of this graph.

19. Given the list of edges and their weights of a weighted undirected connected graph, use Kruskal's algorithm to find a minimum spanning tree of this graph.

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. Display all trees with six vertices.

2. Display a full set of nonisomorphic trees with seven vertices.

***3.** Construct a Huffman code for the letters of the English language based on the frequency of their occurrence in ordinary English text.

4. Compute the number of different spanning trees of K_n for $n = 1, 2, 3, 4, 5, 6$. Conjecture a formula for the number of such spanning trees whenever n is a positive integer.

5. Compare the number of comparisons needed to sort lists of n elements for $n = 100$, 1000, and 10,000 where the elements are randomly selected positive integers, using the selection sort, the insertion sort, the merge sort, and the quick sort.
6. Compute the number of different ways n queens can be arranged on an $n \times n$ chessboard so that no two queens can attack each other for all positive integers n not exceeding 10.

***7.** Find a minimum spanning tree of the graph that connects the capital cities of the fifty states in the U.S. to each other where the weight of each edge is the distance between the cities.

8. Draw the complete game tree for a game of checkers on a 4×4 board.

Writing Projects

RESPOND TO THE FOLLOWING WITH ESSAYS USING OUTSIDE SOURCES.

1. Explain how Cayley used trees to enumerate the number of certain types of hydrocarbons.
2. Define *AVL-trees* (sometimes also known as *height-balanced trees*). Describe how and why AVL-trees are used in a variety of different algorithms.
3. Define *quad trees* and explain how images can be represented using them. Describe how images can be rotated, scaled, and translated by manipulating the corresponding quad tree.
4. Define a *heap* and explain how trees can be turned into heaps. Why are heaps useful in sorting?
5. Describe algorithms for data compression based on letter frequencies, including Huffman coding, and related algorithms based on frequencies of blocks of letters.
6. Discuss how trees are used to model games and to develop winning strategies for games. Explain how to study a game, such as tic-tac-toe, nim, hex, or some other game, using game trees. Be sure to discuss *alpha-beta pruning*.
7. Define the type of graph known as a *mesh of trees*. Explain how this graph is used in applications to very large system integration and parallel computing.
8. Discuss the algorithms used in IP multicasting to avoid loops between routers.
9. Describe an algorithm for finding the minimum spanning tree of a graph so that the maximum degree of any vertex in the spanning tree does not exceed a fixed constant k.
10. Compare and constrast some of the most important sorting algorithms in terms of their complexity and when they are used.
11. Discuss the history and origins of algorithms for constructing minimum spanning trees.
12. Describe algorithms for producing random trees.

Boolean Algebra

9

The circuits in computers and other electronic devices have inputs, each of which is either a 0 or a 1, and produce outputs that are also 0s and 1s. Circuits can be constructed using any basic element that has two different states. Such elements include switches that can be in either the on or the off position and optical devices that can either be lit or unlit. In 1938 Claude Shannon showed how the basic rules of logic, first given by George Boole in 1854 in his *The Laws of Thought*, could be used to design circuits. These rules form the basis for Boolean algebra. In this chapter we develop the basic properties of Boolean algebra. The operation of a circuit is defined by a Boolean function that specifies the value of an output for each set of inputs. The first step in constructing a circuit is to represent its Boolean function by an expression built up using the basic operations of Boolean algebra. We will provide an algorithm for producing such expressions. The expression that we obtain may contain many more operations than are necessary to represent the function. Later in the chapter we will describe methods for finding an expression with the minimum number of sums and products that represents a Boolean function. The procedures that we will develop, Karnaugh maps and the Quine–McCluskey method, are important in the design of efficient circuits.

9.1

Boolean Functions

INTRODUCTION

Boolean algebra provides the operations and the rules for working with the set $\{0, 1\}$. Electronic and optical switches can be studied using this set and the rules of Boolean algebra. The three operations in Boolean algebra that we will use most are complementation, the Boolean sum, and the Boolean product. The **complement** of an element, denoted with a bar, is defined by $\bar{0} = 1$ and $\bar{1} = 0$. The Boolean sum, denoted by $+$ or by *OR*, has the following values:

$$1 + 1 = 1, \qquad 1 + 0 = 1, \qquad 0 + 1 = 1, \qquad 0 + 0 = 0.$$

The Boolean product, denoted by $\cdot$ or by *AND*, has the following values:

$$1 \cdot 1 = 1, \qquad 1 \cdot 0 = 0, \qquad 0 \cdot 1 = 0, \qquad 0 \cdot 0 = 0.$$

When there is no danger of confusion, the symbol $\cdot$ can be deleted, just as in writing algebraic products. Unless parentheses are used, the rules of precedence for Boolean operators are: first, all complements are computed, followed by Boolean products, followed by all Boolean sums. This is illustrated in Example 1.

EXAMPLE 1 Find the value of $1 \cdot 0 + (0 + 1)$.

Solution: Using the definitions of complementation, the Boolean sum, and the Boolean product, it follows that

$$\begin{aligned}(1 \cdot 0) + (0 + 1) &= 0 + 1\\ &= 0 + 0\\ &= 0.\end{aligned}$$

■

The complement, Boolean sum, and Boolean product correspond to the logical operators, ¬, ∨, and ∧, respectively, where 0 corresponds to F (false) and 1 corresponds to T (true). The results of Boolean algebra can be directly translated into results about propositions. Conversely, results about propositions can be translated into statements about Boolean algebra.

BOOLEAN EXPRESSIONS AND BOOLEAN FUNCTIONS

Let $B = \{0, 1\}$. The variable x is called a **Boolean variable** if it assumes values only from B. A function from B^n, the set $\{(x_1, x_2, \ldots, x_n) \mid x_i \in B, 1 \le i \le n\}$, to B is called a **Boolean function of degree** n. The values of a Boolean function are often displayed in tables. For instance, the Boolean function $F(x, y)$ with the value 1 when $x = 1$ and $y = 0$ and the value 0 for all other choices of x and y can be represented by Table 1.

TABLE 1

x	y	$F(x, y)$
1	1	0
1	0	1
0	1	0
0	0	0

Boolean functions can be represented using expressions made up from the variables and Boolean operations. The **Boolean expressions** in the variables $x_1, x_2, \ldots, x_n$ are defined recursively as follows:

$0, 1, x_1, x_2, \ldots, x_n$ are Boolean expressions;

if E_1 and E_2 are Boolean expressions, then $\overline{E}_1$, (E_1E_2), and $(E_1 + E_2)$ are Boolean expressions.

Each Boolean expression represents a Boolean function. The values of this function are obtained by substituting 0 and 1 for the variables in the expression. In Section 9.2 we will show that every Boolean function can be represented by a Boolean expression.

---

web

Claude Elwood Shannon (born 1916). Claude Shannon was born in Gaylord, Michigan, and attended the University of Michigan, graduating in 1936. He continued his studies at M.I.T., where he took the job of maintaining the differential analyzer, a mechanical computing device consisting of shafts and gears built by his professor, Vannevar Bush. Shannon's master's thesis, written in 1936, studied the logical aspects of the differential analyzer. This master's thesis presents the first application of Boolean algebra to the design of switching circuits; it is perhaps the most famous master's thesis of the twentieth century. He received his Ph.D. from M.I.T. in 1940. Shannon joined Bell Laboratories in 1940, where he worked on transmitting data efficiently. He was one of the first people to use bits to represent information. At Bell Laboratories he worked on determining the amount of traffic that telephone lines can carry. Shannon made many fundamental contributions to information theory. In the early 1950s he was one of the founders of the study of artificial intelligence. He joined the M.I.T. faculty in 1956, where he continued his study of information theory.

Shannon has an unconventional side. He is credited with inventing the rocket-powered Frisbee. He is also famous for riding a unicycle down the hallways of Bell Laboratories while juggling four balls. Shannon retired when he was 50 years old, publishing papers sporadically over the following 10 years. Currently, he concentrates on enjoying life and working on some pet projects, such as building a motorized pogo stick. One interesting quote from Shannon, published in *Omni Magazine* in 1987, is "I visualize a time when we will be to robots what dogs are to humans. And I am rooting for the machines."

TABLE 2

x	y	z	xy	$\bar{z}$	$F(x,y,z) = xy + \bar{z}$
1	1	1	1	0	1
1	1	0	1	1	1
1	0	1	0	0	0
1	0	0	0	1	1
0	1	1	0	0	0
0	1	0	0	1	1
0	0	1	0	0	0
0	0	0	0	1	1

EXAMPLE 2 Find the values of the Boolean function represented by $F(x, y, z) = xy + \bar{z}$.

Solution: The values of this function are displayed in Table 2. ■

The Boolean functions F and G of n variables are equal if and only if $F(b_1, b_2, \ldots, b_n) = G(b_1, b_2, \ldots, b_n)$ whenever $b_1, b_2, \ldots, b_n$ belong to B. Two different Boolean expressions that represent the same function are called **equivalent.** For instance, all the Boolean expressions xy, $xy + 0$, and $xy \cdot 1$ are equivalent. The **complement** of the Boolean function F is the function $\overline{F}$, where $\overline{F}(x_1, \ldots, x_n) = \overline{F(x_1, \ldots, x_n)}$. Let F and G be Boolean functions of degree n. The **Boolean sum** $F + G$ and **Boolean product** FG are defined by

$$(F + G)(x_1, \ldots, x_n) = F(x_1, \ldots, x_n) + G(x_1, \ldots, x_n),$$

$$(FG)(x_1, \ldots, x_n) = F(x_1, \ldots, x_n)G(x_1, \ldots, x_n).$$

A Boolean function of degree 2 is a function from a set with four elements, namely, pairs of elements from $B = \{0, 1\}$, to B, a set with two elements. Hence, there are 16 different Boolean functions of degree 2. In Table 3 we display the values of the 16 different Boolean functions of degree 2, labeled $F_1, F_2, \ldots, F_{16}$.

TABLE 3 The Boolean Functions of Degree 2.

x	y	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}
1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
1	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
0	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

TABLE 4 The Number of Boolean Functions of Degree n.

Degree	*Number*
1	4
2	16
3	256
4	65,536
5	4,294,967,296
6	18,446,744,073,709,551,616

EXAMPLE 3 How many different Boolean functions of degree n are there?

Solution: From the product rule for counting, it follows that there are 2^n different n-tuples of 0s and 1s. Since a Boolean function is an assignment of 0 or 1 to each of these 2^n different n-tuples, the product rule shows that there are 2^{2^n} different Boolean functions. ■

Table 4 displays the number of different Boolean functions of degrees 1 through 6. The number of such functions grows extremely rapidly.

IDENTITIES OF BOOLEAN ALGEBRA

There are many identities in Boolean algebra. The most important of these are displayed in Table 5. (The reader should compare these identities to the logical equivalences in Table 5 of Section 1.2 and the set identities in Table 1 in Section 1.5. All are special cases of the same set of identities in a more abstract structure.) These identities are particularly useful in simplifying the design of circuits. Each of the identities in Table 5 can be proved using a table. We will prove one of the distributive laws in this way in the following example. The proofs of the remaining properties are left as exercises for the reader.

EXAMPLE 4 Show that the distributive law $x(y + z) = xy + xz$ is valid.

Solution: The verification of this identity is shown in Table 6. The identity holds because the last two columns of the table agree. ■

The identities in Table 5 can be used to prove further identities. We demonstrate this in the following example.

TABLE 5 Boolean Identities.

Identity	*Name*
$\bar{\bar{x}} = x$	Law of the double complement
$x + x = x$ $x \cdot x = x$	Idempotent laws
$x + 0 = x$ $x \cdot 1 = x$	Identity laws
$x + 1 = 1$ $x \cdot 0 = 0$	Dominance laws
$x + y = y + x$ $xy = yx$	Commutative laws
$x + (y + z) = (x + y) + z$ $x(yz) = (xy)x$	Associative laws
$x + yz = (x + y)(x + z)$ $x(y + z) = xy + xz$	Distributive laws
$\overline{(xy)} = \bar{x} + \bar{y}$ $\overline{(x + y)} = \bar{x}\bar{y}$	De Morgan's laws

TABLE 6

x	y	z	$y + z$	xy	xz	$x(y + z)$	$xy + xz$
1	1	1	1	1	1	1	1
1	1	0	1	1	0	1	1
1	0	1	1	0	1	1	1
1	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0

EXAMPLE 5 Prove the **absorption law** $x(x+y)=x$ using the identities of Boolean algebra. (This is called an absorption law since absorbing $x+y$ into x leaves x unchanged.)

Solution: The steps used to derive this identity and the law used in each step follow:

$x(x+y)=(x+0)(x+y)$	Identity law for the Boolean sum
$=x+0\cdot y$	Distributive law of the Boolean sum over the Boolean product
$=x+y\cdot 0$	Commutative law for the Boolean product
$=x+0$	Dominance law for the Boolean product
$=x$	Identity law for the Boolean sum ■

DUALITY

The identities in Table 5 come in pairs (except for the law of the double complement). To explain the relationship between the two identities in each pair we use the concept of a dual. The **dual** of a Boolean expression is obtained by interchanging Boolean sums and Boolean products and interchanging 0s and 1s.

EXAMPLE 6 Find the duals of $x(y+0)$ and $x\cdot 1+(y+z)$.

Solution: Interchanging $\cdot$ signs and + signs and interchanging 0s and 1s in these expressions produces their duals. The duals are $x+(y\cdot 1)$ and $(x+0)(yz)$, respectively. ■

The dual of a Boolean function F represented by a Boolean expression is the function represented by the dual of this expression. This dual function, denoted by F^d, does not depend on the particular Boolean expression used to represent F. An identity between functions represented by Boolean expressions remains valid when the duals of both sides of the identity are taken. (See Exercise 22 for the reason this is true.) This result, called the **duality principle**, is useful for obtaining new identities.

EXAMPLE 7 Construct an identity from the absorption law $x(x+y)=x$ given in Example 5 by taking duals.

Solution: Taking the duals of both sides of this identity produces the identity $x+xy=x$, which is also called an absorption law. ■

THE ABSTRACT DEFINITION OF A BOOLEAN ALGEBRA

In this section we have focused on Boolean functions and expressions. However, the results we have established can be translated into results about propositions or results about sets. Because of this, it is useful to define Boolean algebras abstractly. Once it is shown that a particular structure is a Boolean algebra, then all results established about Boolean algebras in general apply to this particular structure.

Boolean algebras can be defined in several ways. The most common way is to specify the properties that operations must satisfy, as is done in the following definition.

DEFINITION 1. A **Boolean algebra** is a set B with two binary operations $\vee$ and $\wedge$, elements 0 and 1, and a unary operation $\bar{\ }$ such that the following properties hold for all x, y, and z in B:

$$\left.\begin{aligned} x\vee 0 &= x\\ x\wedge 1 &= x\end{aligned}\right\}\qquad \text{Identity laws}$$

$$\left.\begin{aligned} x\vee \bar{x} &= 1\\ x\wedge \bar{x} &= 0\end{aligned}\right\}\qquad \text{Domination laws}$$

$$\left.\begin{aligned} (x\vee y)\vee z &= x\vee (y\vee z)\\ (x\wedge y)\wedge z &= x\wedge (y\wedge z)\end{aligned}\right\}\qquad \text{Associative laws}$$

$$\left.\begin{aligned} x\vee y &= y\vee x\\ x\wedge y &= y\wedge x\end{aligned}\right\}\qquad \text{Commutative laws}$$

$$\left.\begin{aligned} x\vee (y\wedge z) &= (x\vee y)\wedge (x\vee z)\\ x\wedge (y\vee z) &= (x\wedge y)\vee (x\wedge z)\end{aligned}\right\}\qquad \text{Distributive laws}$$

Using the laws given in Definition 1, it is possible to prove many other laws that hold for every Boolean algebra, such as idempotent and dominance laws. (See Exercises 25–32.)

From our previous discussion, $B = \{0, 1\}$ with the *OR* and *AND* operations and the complement operator, satisfies all these properties. The set of propositions in n variables, with the $\vee$ and $\wedge$ operators, **F** and **T**, and the negation operator, also satisfies all the properties of a Boolean algebra, as can be seen from Table 5 in Section 1.2. Similarly, the set of subsets of a universal set U with the union and intersection operations, the empty set and the universal set, and the set complementation operator, is a Boolean algebra as can be seen by consulting Table 1 in Section 1.5. So, to establish results about each of Boolean expressions, propositions, and sets, we need only prove results about abstract Boolean algebras.

Boolean algebras may also be defined using the notion of a lattice, discussed in Chapter 6. Recall that a lattice L is a partially ordered set in which every pair of elements x, y has a least upper bound, denoted by $\text{lub}(x, y)$ and a greatest lower bound denoted by $\text{glb}(x, y)$. Given two elements x and y of L, we can define two operations $\vee$ and $\wedge$ on pairs of elements of L by $x\vee y = \text{lub}(x, y)$ and $x\wedge y = \text{glb}(x, y)$.

For a lattice L to be a Boolean algebra as specified in Definition 1, it must have two properties. First, it must be **complemented.** For a lattice to be complemented it must have a least element 0 and a greatest element 1 and for every element x of the lattice there must exist an element $\bar{x}$ such that $x\vee \bar{x} = 1$ and $x\wedge \bar{x} = 0$. Second, it must be **distributive.** This means that for every x, y, and z in L, $x\vee(y\wedge z) = (x\vee y)\wedge(x\vee z)$ and $x\wedge(y\vee z) = (x\wedge y)\vee(x\wedge z)$. Showing that a complemented, distributive lattice is a Boolean algebra is left as Exercise 33 at the end of this section.

Exercises

1. Find the values of the following expressions.

a) $1\cdot 0$ **b)** $1+1$ **c)** $\bar{0}\cdot 0$ **d)** $\overline{(1+0)}$

2. Find the values, if any, of the Boolean variable x that satisfy the following equations.

a) $x\cdot 1 = 0$ **b)** $x+x = 0$

c) $x\cdot 1 = x$ **d)** $x\cdot \bar{x} = 1$

3. What values of the Boolean variables x and y satisfy $xy = x + y$?
4. How many different Boolean functions are there of degree 7?
5. Prove the **absorption law** $x + xy = x$ using the laws in Table 5.
6. Show that $F(x, y, z) = xy + xz + yz$ has the value 1 if and only if at least two of the variables x, y, and z have the value 1.
7. Show that $x\bar{y} + y\bar{z} + \bar{x}z = \bar{x}y + \bar{y}z + x\bar{z}$.

Exercises 8–15 deal with the Boolean algebra defined by the Boolean sum and Boolean product on $\{0, 1\}$.

8. Verify the law of the double complement.
9. Verify the idempotent laws.
10. Verify the identity laws.
11. Verify the dominance laws.
12. Verify the commutative laws.
13. Verify the associative laws.
14. Verify the first distributive law in Table 5.
15. Verify De Morgan's laws.

The Boolean operator $\oplus$, called the *XOR* operator, is defined by $1 \oplus 1 = 0$, $1 \oplus 0 = 1$, $0 \oplus 1 = 1$, and $0 \oplus 0 = 0$.

16. Simplify the following expressions.
    a) $x \oplus 0$ b) $x \oplus 1$ c) $x \oplus x$ d) $x \oplus \bar{x}$
17. Show that the following identities hold.
    a) $x \oplus y = (x + y)\overline{(xy)}$
    b) $x \oplus y = (x\bar{y}) + (\bar{x}y)$
18. Show that $x \oplus y = y \oplus x$.
19. Prove or disprove the following equalities.
    a) $x \oplus (y \oplus z) = (x \oplus y) \oplus z$
    b) $x + (y \oplus z) = (x + y) \oplus (x + z)$
    c) $x \oplus (y + z) = (x \oplus y) + (x \oplus z)$
20. Find the duals of the following Boolean expressions.
    a) $x + y$ b) $\bar{x}\,\bar{y}$
    c) $x y \bar{z} + \bar{x}\,\bar{y} z$ d) $x\bar{z} + x \cdot 0 + \bar{x} \cdot 1$

*21. Suppose that F is a Boolean function represented by a Boolean expression in the variables $x_1, \ldots, x_n$. Show that $F^d(x_1, \ldots, x_n) = \overline{F(\bar{x}_1, \ldots, \bar{x}_n)}$.

*22. Show that if F and G are Boolean functions represented by Boolean expressions in n variables and $F = G$, then $F^d = G^d$, where F^d and G^d are the Boolean functions represented by the duals of the Boolean expressions representing F and G, respectively. (*Hint:* Use the result of Exercise 21.)

*23. How many different Boolean functions $F(x, y, z)$ are there so that $F(\bar{x}, \bar{y}, \bar{z}) = F(x, y, z)$ for all values of the Boolean variables x, y, and z?

*24. How many different Boolean functions $F(x, y, z)$ are there so that $F(\bar{x}, y, z) = F(x, \bar{y}, z) = F(x, y, \bar{z})$ for all values of the Boolean variables x, y, z?

In Exercises 25–32, use the laws in Definition 1 to show that the stated properties hold in every Boolean algebra.

25. Show that in a Boolean algebra, the **idempotent properties** $x \vee x = x$ and $x \wedge x = x$ hold for every element x.
26. Show that in a Boolean algebra, every element x has a unique complement $\bar{x}$ such that $x \vee \bar{x} = 1$ and $x \wedge \bar{x} = 0$.
27. Show that in a Boolean algebra, the complement of the element 0 is the element 1 and vice versa.
28. Prove that in a Boolean algebra, the **law of the double complement** holds; that is, $\bar{\bar{x}} = x$ for every element x.
29. Show that **De Morgan's laws** hold in a Boolean algebra. That is, show that for all x and y, $\overline{(x \vee y)} = \bar{x} \wedge \bar{y}$ and $\overline{(x \wedge y)} = \bar{x} \vee \bar{y}$.
30. Show that in a Boolean algebra, the **modular properties** hold. That is, show that $x \wedge (y \vee (x \wedge z)) = (x \wedge y) \vee (x \wedge z)$ and $x \vee (y \wedge (x \vee z)) = (x \vee y) \wedge (x \vee z)$.
31. Show that in a Boolean algebra, if $x \vee y = 0$, then $x = 0$ and $y = 0$ and that if $x \wedge y = 1$, then $x = 1$ and $y = 1$.
32. Show that in a Boolean algebra, the **dual** of an identity, obtained by interchanging the $\vee$ and $\wedge$ operators and interchanging the elements 0 and 1, is also a valid identity.
33. Show that a complemented, distributive lattice is a Boolean algebra.

9.2

Representing Boolean Functions

Two important problems of Boolean algebra will be studied in this section. The first problem is: Given the values of a Boolean function, how can a Boolean expression that represents this function be found? This problem will be solved by showing that any Boolean function may be represented by a Boolean sum of Boolean products of the variables and their complements. The solution of this problem shows that every Boolean function can be represented using the three Boolean operators $\cdot$, $+$, and $^{-}$. The second problem is: Is there a smaller set of operators that can be used to represent all Boolean functions? We will answer this problem by showing that all Boolean functions can be

TABLE 1

x	y	z	F	G
1	1	1	0	0
1	1	0	0	1
1	0	1	1	0
1	0	0	0	0
0	1	1	0	0
0	1	0	0	1
0	0	1	0	0
0	0	0	0	0

represented using only one operator. Both of these problems have practical importance in circuit design.

SUM-OF-PRODUCTS EXPANSIONS

We will use examples to illustrate one important way to find a Boolean expression that represents a Boolean function.

EXAMPLE 1 Find Boolean expressions that represent the functions $F(x, y, z)$ and $G(x, y, z)$, which are given in Table 1.

Solution: An expression that has the value 1 when $x = z = 1$ and $y = 0$, and the value 0 otherwise, is needed to represent F. Such an expression can be formed by taking the Boolean product of x, $\bar{y}$, and z. This product, $x\bar{y}z$, has the value 1 if and only if $x = \bar{y} = z = 1$, which holds if and only if $x = z = 1$ and $y = 0$.

To represent G, we need an expression that equals 1 when $x = y = 1$ and $z = 0$, or when $x = z = 0$ and $y = 1$. We can form an expression with these values by taking the Boolean sum of two different Boolean products. The Boolean product $xy\bar{z}$ has the value 1 if and only if $x = y = 1$ and $z = 0$. Similarly, the product $\bar{x}y\bar{z}$ has the value 1 if and only if $x = z = 0$ and $y = 1$. The Boolean sum of these two products, $xy\bar{z} + \bar{x}y\bar{z}$, represents G, since it has the value 1 if and only if $x = y = 1$ and $z = 0$ or $x = z = 0$ and $y = 1$. ■

Example 1 illustrates a procedure for constructing a Boolean expression representing a function with given values. Each combination of values of the variables for which the function has the value 1 leads to a Boolean product of the variables or their complements.

DEFINITION 1. A *literal* is a Boolean variable or its complement. A *minterm* of the Boolean variables $x_1, x_2, \ldots, x_n$ is a Boolean product $y_1 y_2 \cdots y_n$, where $y_i = x_i$ or $y_i = \bar{x}_i$. Hence, a minterm is a product of n literals, with one literal for each variable.

A minterm has the value 1 for one and only one combination of values of its variables. More precisely, the minterm $y_1y_2\cdots y_n$ is 1 if and only if each y_i is 1, and this occurs if and only if $x_i = 1$ when $y_i = x_i$ and $x_i = 0$ when $y_i = \overline{x}_i$.

EXAMPLE 2 Find a minterm that equals 1 if $x_1 = x_3 = 0$ and $x_2 = x_4 = x_5 = 1$, and equals 0 otherwise.

Solution: The minterm $\overline{x}_1x_2\overline{x}_3x_4x_5$ has the correct set of values. ■

By taking Boolean sums of distinct minterms we can build up a Boolean expression with a specified set of values. In particular, a Boolean sum of minterms has the value 1 when exactly one of the minterms in the sum has the value 1. It has the value 0 for all other combinations of values of the variables. Consequently, given a Boolean function, a Boolean sum of minterms can be formed that has the value 1 when this Boolean function has the value 1, and has the value 0 when the function has the value 0. The minterms in this Boolean sum correspond to those combinations of values for which the function has the value 1. The sum of minterms that represents the function is called the **sum-of-products expansion** or the **disjunctive normal form** of the Boolean function.

EXAMPLE 3 Find the sum-of-products expansion for the function $F(x, y, z) = (x + y)\overline{z}$.

Solution: The first step is to find the values of F. These are found in Table 2. The sum-of-products expansion of F is the Boolean sum of three minterms corresponding to the three rows of this table that give the value 1 for the function. This gives

$$F(x, y, z) = xy\overline{z} + x\overline{y}\overline{z} + \overline{x}y\overline{z}.$$ ■

It is also possible to find a Boolean expression that represents a Boolean function by taking a Boolean product of Boolean sums. The resulting expansion is called the **conjunctive normal form** or **product-of-sums expansion** of the function. These expansions can be found from sum-of-products expansions by taking duals. How to find such expansions directly is described in Exercise 10 at the end of this section.

TABLE 2

x	y	z	$x + y$	$\overline{z}$	$(x + y)\overline{z}$
1	1	1	1	0	0
1	1	0	1	1	1
1	0	1	1	0	0
1	0	0	1	1	1
0	1	1	1	0	0
0	1	0	1	1	1
0	0	1	0	0	0
0	0	0	0	1	0

FUNCTIONAL COMPLETENESS

Every Boolean function can be expressed as a Boolean sum of minterms. Each minterm is the Boolean product of Boolean variables or their complements. This shows that every Boolean function can be represented using the Boolean operators $\cdot$, $+$, and $^{-}$. Since every Boolean function can be represented using these operators we say that the set $\{\cdot, +, ^{-}\}$ is **functionally complete**. Can we find a smaller set of functionally complete operators? We can do so if one of the three operators of this set can be expressed in terms of the other two. This can be done using one of De Morgan's laws. We can eliminate all Boolean sums using the identity

$$x + y = \overline{\bar{x}\,\bar{y}},$$

which is obtained by taking complements of both sides in the second De Morgan's law, given in Table 5 in Section 9.1, and then applying the double complementation law. This means that the set $\{\cdot, ^{-}\}$ is functionally complete. Similarly, we could eliminate all Boolean products using the identity

$$xy = \overline{\bar{x} + \bar{y}},$$

which is obtained by taking complements of both sides in the first De Morgan's law, given in Table 5 in Section 9.1, and then applying the double complementation law. Consequently $\{+, ^{-}\}$ is functionally complete. Note that the set $\{+, \cdot\}$ is not functionally complete, since it is impossible to express the Boolean function $F(x) = \bar{x}$ using these operators (see Exercise 19).

We have found sets containing two operators that are functionally complete. Can we find a smaller set of functionally complete operators, namely, a set containing just one operator? Such sets exist. Define two operators, the $\mid$ or *NAND* operator, defined by $1 \mid 1 = 0$ and $1 \mid 0 = 0 \mid 1 = 0 \mid 0 = 1$; and the $\downarrow$ or *NOR* operator, defined by $1 \downarrow 1 = 1 \downarrow 0 = 0 \downarrow 1 = 0$ and $0 \downarrow 0 = 1$. Both of the sets $\{\mid\}$ and $\{\downarrow\}$ are functionally complete. To see that $\{\mid\}$ is functionally complete, since $\{\cdot, ^{-}\}$ is functionally complete, all that we have to do is show that both of the operators $\cdot$ and $^{-}$ can be expressed using just the $\mid$ operator. This can be done as follows:

$$\bar{x} = x \mid x,$$

$$xy = (x \mid y) \mid (x \mid y).$$

The reader should verify these identities (see Exercise 14). We leave the demonstration that $\{\downarrow\}$ is functionally complete for the reader (see Exercises 15 and 16).

Exercises

1. Find a Boolean product of the Boolean variables x, y, and z, or their complements, that has the value 1 if and only if

a) $x = y = 0, z = 1$ **b)** $x = 0, y = 1, z = 0$

c) $x = 0, y = z = 1$ **d)** $x = y = z = 0$

2. Find the sum-of-products expansions of the following Boolean functions.

a) $F(x, y) = \bar{x} + y$ **b)** $F(x, y) = x\bar{y}$

c) $F(x, y) = 1$ **d)** $F(x, y) = \bar{y}$

3. Find the sum-of-products expansions of the following Boolean functions.

a) $F(x, y, z) = x + y + z$

b) $F(x, y, z) = (x + z)y$

c) $F(x, y, z) = x$

d) $F(x, y, z) = x\bar{y}$

4. Find the sum-of-products expansions of the Boolean function $F(x, y, z)$ that equals 1 if and only if

a) $x = 0$. **b)** $xy = 0$.

c) $x + y = 0$. **d)** $xyz = 0$.

5. Find the sum-of-products expansion of the Boolean function $F(w, x, y, z)$ that has the value 1 if and only if an odd number of w, x, y, and z have the value 1.

6. Find the sum-of-products expansion of the Boolean function $F(x_1, x_2, x_3, x_4, x_5)$ that has the value 1 if and only if three or more of the variables x_1, x_2, x_3, x_4, and x_5 have the value 1.

Another way to find a Boolean expression that represents a Boolean function is to form a Boolean product of Boolean sums of literals. Exercises 7–11 are concerned with representations of this kind.

7. Find a Boolean sum containing either x or $\bar{x}$, either y or $\bar{y}$, and either z or $\bar{z}$ that has the value 0 if and only if
   a) $x = y = 1, z = 0$. **b)** $x = y = z = 0$.
   c) $x = z = 0, y = 1$.
8. Find a Boolean product of Boolean sums of literals that has the value 0 if and only if either $x = y = 1$ and $z = 0$, $x = z = 0$ and $y = 1$, or $x = y = z = 0$. [*Hint:* Take the Boolean product of the Boolean sums found in parts (a), (b), and (c) in Exercise 7.]
9. Show that the Boolean sum $y_1 + y_2 + \cdots + y_n$, where $y_i = x_i$ or $y_i = \bar{x}_i$, has the value 0 for exactly one combination of the values of the variables, namely, when $x_i = 0$ if $y_i = x_i$ and $x_i = 1$ if $y_i = \bar{x}_i$. This Boolean sum is called a **maxterm.**
10. Show that a Boolean function can be represented as a Boolean product of maxterms. This representation is called the **product-of-sums expansion or conjunctive normal form** of the function. (*Hint:* Include one maxterm in this product for each combination of the variables where the function has the value 0.)
11. Find the product-of-sums expansion of each of the Boolean functions in Exercise 3.
12. Express each of the following Boolean functions using the operators · and ¯.
    a) $x + y + z$ **b)** $x + y(\bar{x} + z)$
    c) $(x + \bar{y})$ **d)** $\bar{x}(x + y + z)$
13. Express each of the Boolean functions in Exercise 12 using the operators + and ¯.
14. Show that
    a) $\bar{x} = x \mid x$.
    b) $xy = (x \mid y) \mid (x \mid y)$.
    c) $x + y = (x \mid x) \mid (y \mid y)$.
15. Show that
    a) $\bar{x} = x \downarrow x$.
    b) $xy = (x \downarrow x) \downarrow (y \downarrow y)$.
    c) $x + y = (x \downarrow y) \downarrow (x \downarrow y)$.
16. Show that $\{\downarrow\}$ is functionally complete using Exercise 15.
17. Express each of the Boolean functions in Exercise 3 using the operator $\mid$.
18. Express each of the Boolean functions in Exercise 3 using the operator $\downarrow$.
19. Show that the set of operators $\{+, \cdot\}$ is not functionally complete.
20. Are the following sets of operators functionally complete?
    a) $\{+, \oplus\}$ **b)** $\{\bar{\ }, \oplus\}$ **c)** $\{\cdot, \oplus\}$

9.3

Logic Gates

INTRODUCTION

web

Boolean algebra is used to model the circuitry of electronic devices. Each input and each output of such a device can be thought of as a member of the set {0, 1}. A computer, or other electronic device, is made up of a number of circuits. Each circuit can be designed using the rules of Boolean algebra that were studied in Sections 9.1 and 9.2. The basic elements of circuits are called **gates.** Each type of gate implements a Boolean operation. In this section we define several types of gates. Using these gates, we will apply the rules of Boolean algebra to design circuits that perform a variety of tasks. The circuits that we will study in this chapter give output that depends only on the input, and not on the current state of the circuit. In other words, these circuits have no memory capabilities. Such circuits are called **combinational circuits** or **gating networks.**

We will construct combinational circuits using three types of elements. The first is an **inverter,** which accepts the value of one Boolean variable as input and produces the complement of this value as its output. The symbol used for an inverter is shown in Figure 1(a). The input to the inverter is shown on the left side entering the element, and the output is shown on the right side leaving the element.

The next type of element we will use is the *OR* gate. The inputs to this gate are the values of two or more Boolean variables. The output is the Boolean sum of their values.

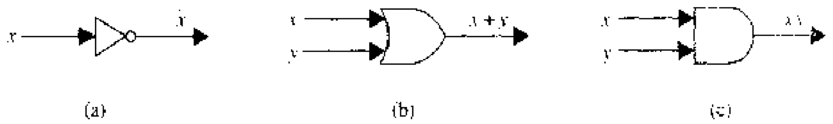


FIGURE 1 Basic Types of Gates.

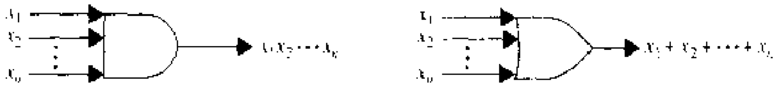


FIGURE 2 Gates with *n* Inputs.

The symbol used for an *OR* gate is shown in Figure 1(b). The inputs to the *OR* gate are shown on the left side entering the element, and the output is shown on the right side leaving the element.

The third type of element we will use is the *AND* gate. The inputs to this gate are the values of two or more Boolean variables. The output is the Boolean product of their values. The symbol used for an *AND* gate is shown in Figure 1(c). The inputs to the *AND* gate are shown on the left side entering the element, and the output is shown on the right side leaving the element.

We will permit multiple inputs to *AND* and *OR* gates. The inputs to each of these gates are shown on the left side entering the element, and the output is shown on the right side. Examples of *AND* and *OR* gates with *n* inputs are shown in Figure 2.

COMBINATIONS OF GATES

Combinational circuits can be constructed using a combination of inverters, *OR* gates, and *AND* gates. When combinations of circuits are formed, some gates may share inputs. This is shown in one of two ways in depictions of circuits. One method is to use branchings that indicate all the gates that use a given input. The other method is to indicate this input separately for each gate. Figure 3 illustrates the two ways of showing

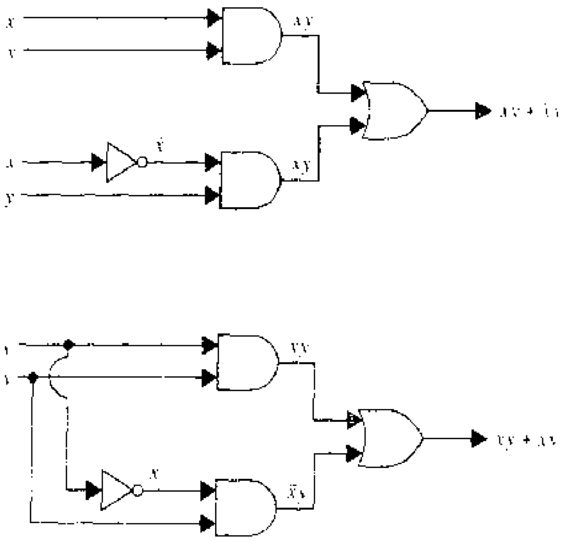


FIGURE 3 Two Ways to Draw the Same Circuit.

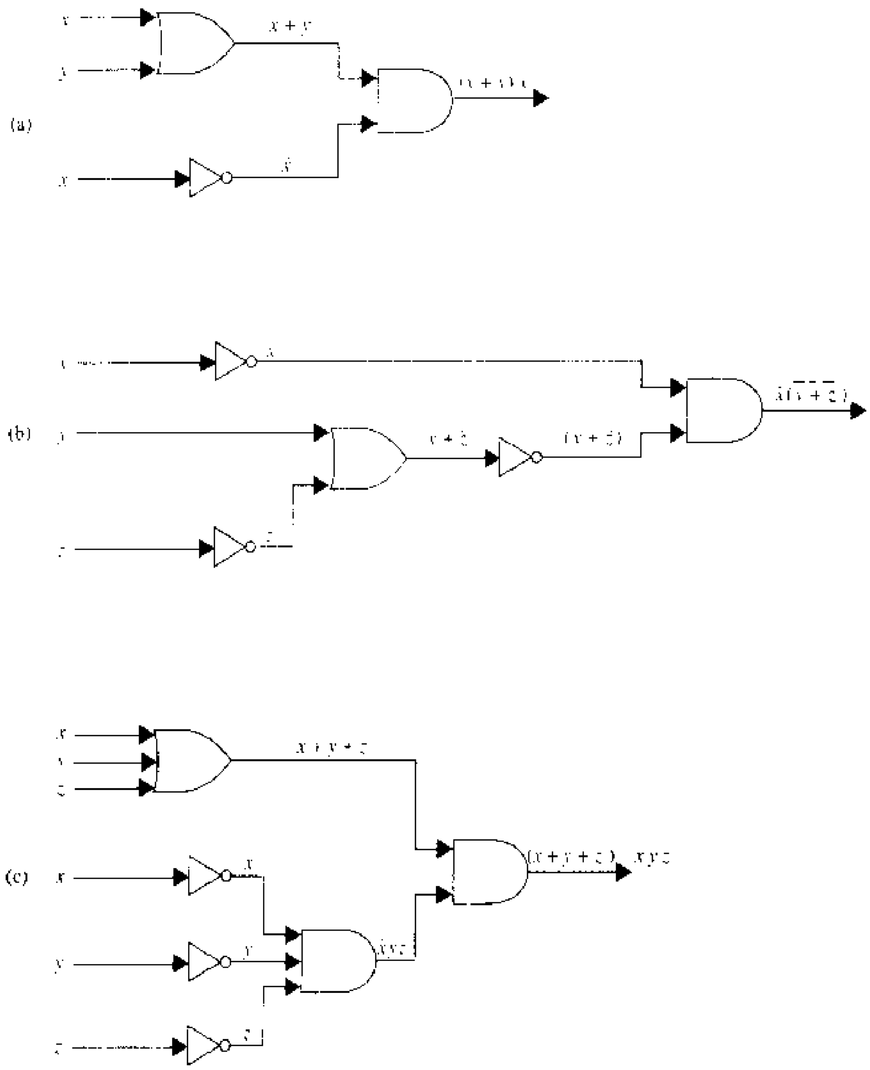

(a)
x
y
x + y
$(x + y)\bar{x}$
x
$\bar{x}$
(b)
x
$\bar{x}$
$\bar{x}(\overline{y + \bar{z}})$
y
$y + \bar{z}$
$(\overline{y + \bar{z}})$
z
$\bar{z}$
x
y
z
x + y + z
(c)
x
$\bar{x}$
$(x + y + z)\bar{x}\bar{y}\bar{z}$
y
$\bar{y}$
$\bar{x}\bar{y}\bar{z}$
z
$\bar{z}$

FIGURE 4 **Circuits That Produce the Outputs Specified in Example 1.**

gates with the same input values. Note also that output from a gate may be used as input by one or more other elements, as show in Figure 3. Both drawings in Figure 3 depict the circuit that produces the output $xy + \bar{x}y$.

EXAMPLE 1 Construct circuits that produce the following outputs: (a) $(x + y)\bar{x}$, (b) $\bar{x}\,\overline{(y + \bar{z})}$, and (c) $(x + y + z)(\bar{x}\bar{y}\bar{z})$.

Solution: Circuits that produce these outputs are shown in Figure 4. ■

EXAMPLES OF CIRCUITS

We will give some examples of circuits that perform some useful functions.

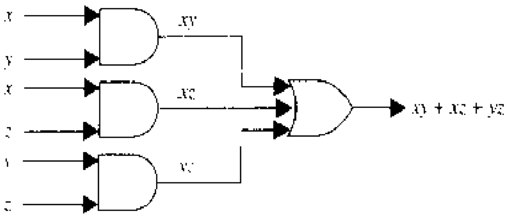


FIGURE 5 **A Circuit for Majority Voting.**

EXAMPLE 2 A committee of three individuals decides issues for an organization. Each individual votes either yes or no for each proposal that arises. A proposal is passed if it receives at least two yes votes. Design a circuit that determines whether a proposal passes.

Solution: Let $x = 1$ if the first individual votes yes, and $x = 0$ if this individual votes no; let $y = 1$ if the second individual votes yes, and $y = 0$ if this individual votes no; let $z = 1$ if the third individual votes yes, and $z = 0$ if this individual votes no. Then a circuit must be designed that produces the output 1 from the inputs x, y, and z when two or more of x, y, and z are 1. One representation of the Boolean function that has these output values is $xy + xz + yz$ (see Exercise 6 in Section 9.1). The circuit that implements this function is shown in Figure 5. ■

EXAMPLE 3 Sometimes light fixtures are controlled by more than one switch. Circuits need to be designed so that flipping any one of the switches for the fixture turns the light on when it is off and turns the light off when it is on. Design circuits that accomplish this when there are two switches and when there are three switches.

TABLE 1

x	y	$F(x, y)$
1	1	1
1	0	0
0	1	0
0	0	1

Solution: We will begin by designing the circuit that controls the light fixture when two different switches are used. Let $x = 1$ when the first switch is closed and $x = 0$ when it is open, and let $y = 1$ when the second switch is closed and $y = 0$ when it is open. Let $F(x, y) = 1$ when the light is on and $F(x, y) = 0$ when it is off. We can arbitrarily decide that the light will be on when both switches are closed, so that $F(1, 1) = 1$. This determines all the other values of F. When one of the two switches is opened, the light goes off, so $F(1, 0) = F(0, 1) = 0$. When the other switch is also opened, the light goes on, so that $F(0, 0) = 1$. Table 1 displays these values. We see that $F(x, y) = xy + \bar{x}\bar{y}$. This function is implemented by the circuit shown in Figure 6.

We will now design a circuit for three switches. Let x, y, and z be the Boolean variables that indicate whether each of the three switches is closed. We let $x = 1$ when the first switch is closed, and $x = 0$ when it is open; $y = 1$ when the second switch is closed, and $y = 0$ when it is open; and $z = 1$ when the third switch is closed, and

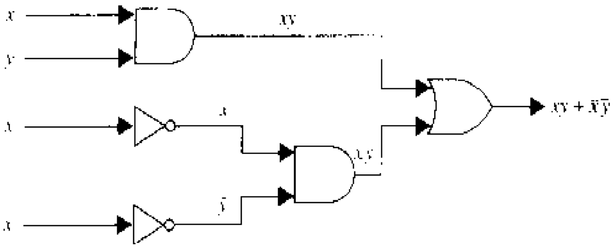


FIGURE 6 **A Circuit for a Light Controlled by Two Switches.**

TABLE 2

x	y	z	$F(x, y, z)$
1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	1
0	0	1	1
0	0	0	0

$z = 0$ when it is open. Let $F(x, y, z) = 1$ when the light is on and $F(x, y, z) = 0$ when the light is off. We can arbitrarily specify that the light be on when all three switches are closed so that $F(1, 1, 1) = 1$. This determines all other values of F. When one switch is opened, the light goes off so that $F(1, 1, 0) = F(1, 0, 1) = F(0, 1, 1) = 0$. When a second switch is opened, the light goes on so that $F(1, 0, 0) = F(0, 1, 0) = F(0, 0, 1) = 1$. Finally, when the third switch is opened, the light goes off again so that $F(0, 0, 0) = 0$. Table 2 shows the values of this function.

The function F can be represented by its sum-of-products expansion so that $F(x, y, z) = xyz + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}z$. The circuit shown in Figure 7 implements this function. ■

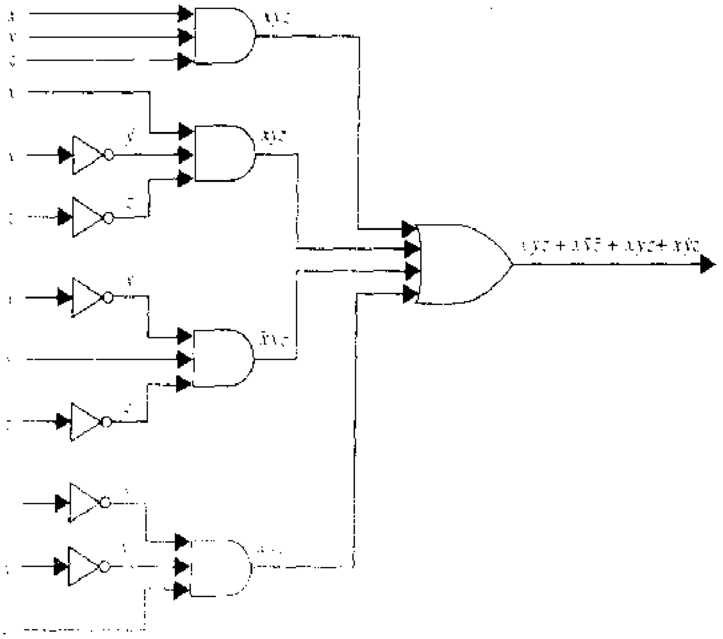

x
y
z
xyz
x
y
$\bar{y}$
z
$\bar{z}$
$x\bar{y}\bar{z}$
x
$\bar{x}$
y
z
$\bar{z}$
$\bar{x}y\bar{z}$
$\bar{x}$
y
$\bar{y}$
z
$\bar{x}\bar{y}z$
$xyz + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}z$

FIGURE 7 A Circuit for a Fixture Controlled by Three Switches.

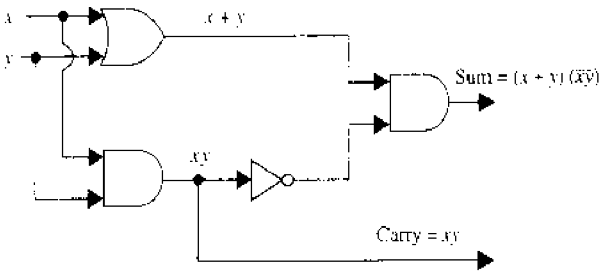


FIGURE 8 **The Half Adder.**

ADDERS

web

We will illustrate how logic circuits can be used to carry out addition of two positive integers from their binary expansions. We will build up the circuitry to do this addition from some component circuits. First, we will build a circuit that can be used to find $x + y$, where x and y are two bits. The input to our circuit will be x and y, since these each have the value 0 or the value 1. The output will consist of two bits, namely, s and c, where s is the sum bit and c is the carry bit. This circuit is called a **multiple output circuit** since it has more than one output. The circuit that we are designing is called the **half adder**, since it adds two bits, without considering a carry from a previous addition. We show the input and output for the half adder in Table 3. From Table 3 we see that $c = xy$ and that $s = xy + xy = (x + y)(\overline{xy})$. Hence, the circuit shown in Figure 8 computes the sum bit s and the carry bit c from the bits x and y.

We use the **full adder** to compute the sum bit and the carry bit when two bits and a carry are added. The inputs to the full adder are the bits x and y and the carry c_i. The outputs are the sum bit s and the new carry c_{i+1}. The inputs and outputs for the full adder are shown in Table 4.

TABLE 3 **Input and Output for the Half Adder.**

Input		*Output*	
x	y	s	c
1	1	0	1
1	0	1	0
0	1	1	0
0	0	0	0

TABLE 4 **Input and Output for the Full Adder.**

Input			*Output*	
x	y	c_i	s	c_{i+1}
1	1	1	1	1
1	1	0	0	1
1	0	1	0	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0

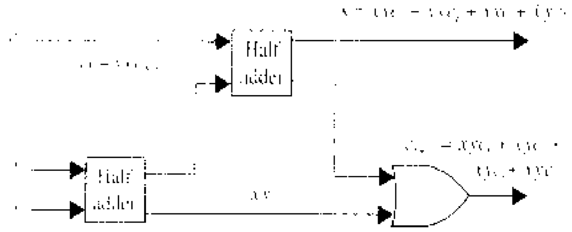


FIGURE 9 A Full Adder.

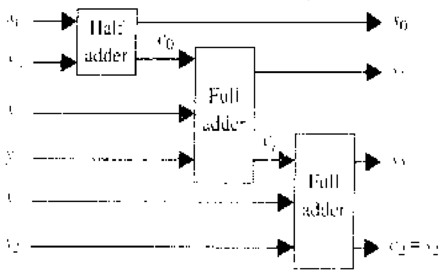


FIGURE 10 Adding Two Three-Bit Integers with Full and Half Adders.

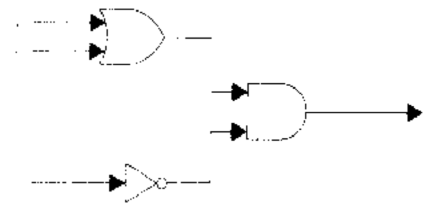
The two outputs of the full adder, the sum bit s and the carry c_{i+1}, are given by the sum-of-products expansions $xyc_i + x\bar{y}\bar{c}_i + \bar{x}y\bar{c}_i + \bar{x}\bar{y}c_i$ and $xyc_i + xy\bar{c}_i + x\bar{y}c_i + \bar{x}yc_i$, respectively. However, instead of designing the full adder from scratch, we will use half adders to produce the desired output. A full adder circuit using half adders is shown in Figure 9.

Finally, in Figure 10 we show how full and half adders can be used to add the two three-bit integers $(x_2x_1x_0)_2$ and $(y_2y_1y_0)_2$ to produce the sum $(s_3s_2s_1s_0)_2$. Note that s_3, the highest-order bit in the sum, is given by the carry c_2.

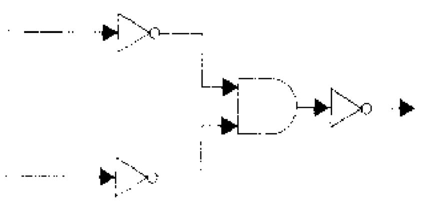
Exercises

In Exercises 1–5 find the output of the given circuit.

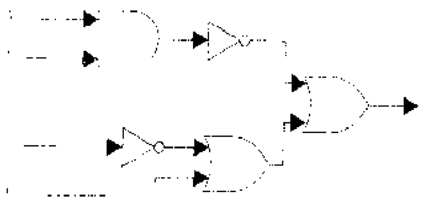
1.



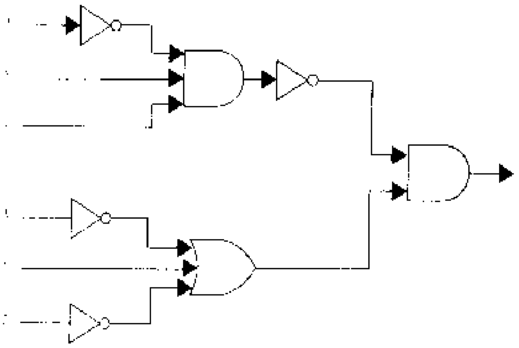
2.



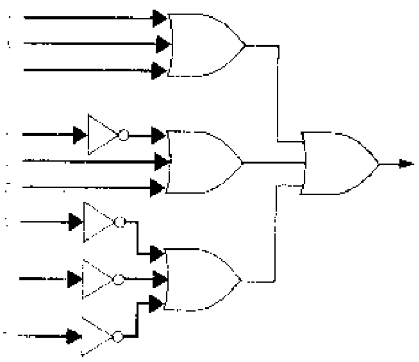
3.



4.

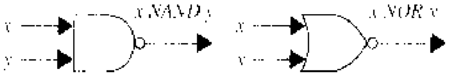


5.



6. Construct circuits from inverters, *AND* gates, and *OR* gates to produce the following outputs.
   a) $x + y$ b) $\overline{(x + y)}x$
   c) $xyz + x\bar{y}z$ d) $\overline{(x + z)}(\bar{y} + z)$
7. Design a circuit that implements majority voting for five individuals.
8. Design a circuit for a light fixture controlled by four switches where flipping one of the switches turns the light on when it is off and turns it off when it is on.
9. Show how the sum of two five-bit integers can be found using full and half adders.
10. Construct a circuit for a half subtractor using *AND* gates, *OR* gates, and inverters. A **half subtractor** has two bits as input and produces as output a difference bit and a borrow.
11. Construct a circuit for a full subtractor using *AND* gates, *OR* gates, and inverters. A **full subtractor** has two bits and a borrow as input, and produces as output a difference bit and a borrow.
12. Use the circuits from Exercises 10 and 11 to find the difference of two four-bit integers, where the first integer is greater than the second integer.
*13. Construct a circuit that compares the two-bit integers $(x_1x_0)_2$ and $(y_1y_0)_2$, returning an output of 1 when the first of these numbers is larger and an output of 0 otherwise.
*14. Construct a circuit that computes the product of the two-bit integers $(x_1x_0)_2$ and $(y_1y_0)_2$. The circuit should have four output bits for the bits in the product.

Two gates that are often used in circuits are *NAND* and *NOR* gates. When *NAND* or *NOR* gates are used to represent circuits, no other types of gates are needed. The notation for these gates is as follows:



*15. Use *NAND* gates to construct circuits with the following outputs.
   a) x b) $x + y$ c) xy d) $x \oplus y$
*16. Use *NOR* gates to construct circuits for the outputs given in Exercise 15.
*17. Construct a half adder using *NAND* gates.
*18. Construct a half adder using *NOR* gates.

A **multiplexer** is a switching circuit that produces as output one of a set of input bits based on the value of control bits.

19. Construct a multiplexer using *AND* gates, *OR* gates, and inverters that has as input the four bits x_0, x_1, x_2, and x_3 and the two control bits c_0 and c_1. Set up the circuit so that x_i is the output where i is the value of the two-bit integer $(c_1c_0)_2$.

9.4
Minimization of Circuits

INTRODUCTION

The efficiency of a combinational circuit depends on the number and arrangement of its gates. The process of designing a combinational circuit begins with the table specifying the output for each combination of input values. We can always use the sum-of-products expansion of a circuit to find a set of logic gates that will implement this circuit. However, the sum-of-products expansion may contain many more terms than are necessary. Terms in a sum-of-products expansion that differ in just one variable, so that in one term this variable occurs and in the other term the complement of this variable occurs, can be combined. For instance, consider the circuit that has output 1 if and only if $x = y = z = 1$ or $x = z = 1$ and $y = 0$. The sum-of-products expansion of this circuit is $xyz + xyz$. The two products in this expansion differ in exactly one variable, namely, y. They can be combined as follows:

$$\begin{aligned} xyz + xyz &= (y + y)(xz) \\ &= 1 \cdot (xz) \\ &= xz \end{aligned}$$

Hence, xz is a Boolean expression with fewer operators that represents the circuit. We show two different implementations of this circuit in Figure 1. The second circuit uses only one gate, whereas the first circuit uses three gates and an inverter.

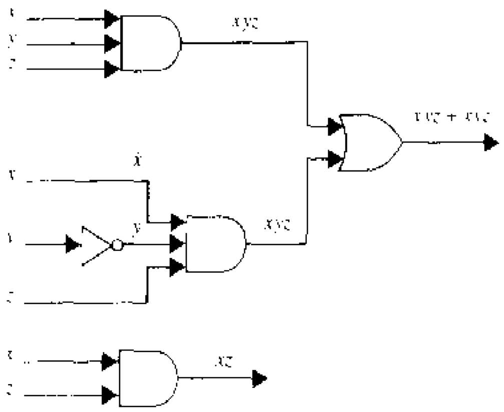


FIGURE 1 Two Circuits with the Same Output.

This example shows that combining terms in the sum-of-products expansion of a circuit leads to a simpler expression for the circuit. We will describe two procedures that simplify sum-of-products expansions. The goal of both of these procedures is to produce Boolean sums of Boolean products that contain the least number of products of literals such that these products contain the least number of literals possible among all sums of products that represent a Boolean function.

The techniques described in this section for simplifying sum-of-product expansions are still of practical value. However, modern circuits are often built using more complicated types of elements than *AND* gates, *OR* gates, and inverters. Various procedures are used to simplify circuits built using these more complicated elements. However, many of these methods use ideas similar to those described in this section.

KARNAUGH MAPS

web

To reduce the number of terms in a Boolean expression representing a circuit, it is necessary to find terms to combine. There is a graphical method, called a **Karnaugh map**, for finding terms to combine for Boolean functions involving a relatively small number of variables. The method we will describe was introduced by Maurice Karnaugh in 1953. His method is based on earlier work by E. W. Veitch. (This method is usually applied only when the function involves six or fewer variables.) Karnaugh maps give us a visual method for simplifying sum-of-products expansions; they are not suited for mechanizing this process. We will first illustrate how Karnaugh maps are used to simplify expansions of Boolean functions in two variables.

There are four possible minterms in the sum-of-products expansion of a Boolean function in the two variables x and y. A Karnaugh map for a Boolean function in these two variables consists of four squares, where a 1 is placed in the square representing

---

Maurice Karnaugh (born 1924). Maurice Karnaugh, born in New York City, received his B.S. from the City College of New York and his M.S. and Ph.D. from Yale University. He was a member of the technical staff at Bell Laboratories from 1952 until 1966 and Manager of Research and Development at the Federal Systems Division of AT&T from 1966 to 1970. In 1970 he joined IBM as a member of the research staff. Karnaugh has made fundamental contributions to the application of digital techniques in both computing and telecommunications. His current interests include knowledge-based systems in computers and heuristic search methods.

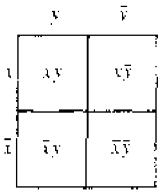


FIGURE 2 **Karnaugh Maps in Two Variables.**

a minterm if this minterm is present in the expansion. Squares are said to be **adjacent** if the minterms that they represent differ in exactly one literal. For instance the square representing xy is adjacent to the squares representing $x\bar{y}$ and $\bar{x}y$. The four squares and the terms that they represent are shown in Figure 2.

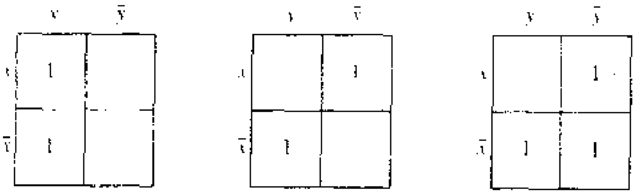
EXAMPLE 1 Find the Karnaugh maps for (a) $xy + \bar{x}y$, (b) $x\bar{y} + \bar{x}y$, and (c) $x\bar{y} + \bar{x}y + \bar{x}\bar{y}$.

Solution: We include a 1 in a square when the minterm represented by this square is present in the sum-of-products expansion. The three Karnaugh maps are shown in Figure 3. ■

We can identify minterms that can be combined from the Karnaugh map. Whenever there are 1s in two adjacent squares in the Karnaugh map, the minterms represented by these squares can be combined into a product involving just one of the variables. For instance, $x\bar{y}$ and $\bar{x}\bar{y}$ are represented by adjacent squares and can be combined into $\bar{y}$, since $x\bar{y} + \bar{x}\bar{y} = (x + \bar{x})\bar{y} = \bar{y}$. Moreover, if 1s are in all four squares, the four minterms can be combined into one term, namely, the Boolean expression 1 that involves none of the variables. We circle blocks of squares in the Karnaugh map that represent minterms that can be combined and then find the corresponding sum of products. The goal is to identify the largest possible blocks, and to cover all the 1s with the fewest blocks using the largest blocks first and always using the largest possible blocks.

EXAMPLE 2 Simplify the sum-of-products expansions given in Example 1.

Solution: The grouping of minterms is shown in Figure 4 using the Karnaugh maps for these expansions. Minimal expansions for these sums-of-products are (a) y, (b) $x\bar{y} + \bar{x}y$, and (c) $\bar{x} + \bar{y}$. ■



(a) (b) (c)

FIGURE 3 **Karnaugh Maps for the Sum-of-Products Expansions in Example 1.**

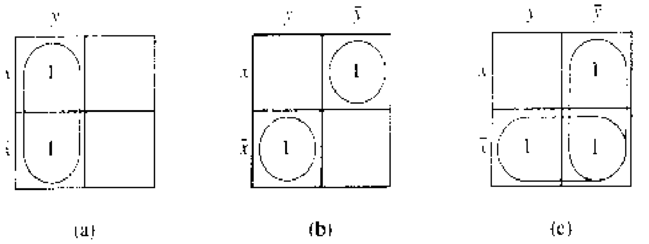


FIGURE 4 **Simplifying the Sum-of-Products Expansion from Example 1.**

A Karnaugh map in three variables is a rectangle divided into eight squares. The squares represent the eight possible minterms in three variables. Two squares are said to be adjacent if the minterms that they represent differ in exactly one literal. One of the ways to form a Karnaugh map in three variables is shown in Figure 5(a). This Karnaugh map can be thought of as lying on a cylinder, as shown in Figure 5(b). On the cylinder two squares have a common border if and only if they are adjacent.

To simplify a sum-of-products expansion in three variables, we use the Karnaugh map to identify blocks of minterms that can be combined. Blocks of two adjacent squares represent pairs of minterms that can be combined into a product of two literals; 2 × 2 and 4 × 1 blocks of squares represent minterms that can be combined into a single literal; and the block of all eight squares represents a product of no literals, namely, the function **1**. In Figure 6, 1 × 2, 2 × 1, 2 × 2, 4 × 1, and 4 × 2 blocks and the products they represent are shown.

The goal is to identify the largest possible blocks in the map and cover all the 1s in the map with the least number of blocks, using the largest blocks first. The largest possible blocks are always chosen. Note that there may be more than one way to do this. The following example illustrates how Karnaugh maps in three variables are used.

EXAMPLE 3 Use Karnaugh maps to simplify the sum-of-products expansions (a) $xy\bar{z} + x\bar{y}\bar{z} + \bar{x}yz + \bar{x}\bar{y}\bar{z}$, (b) $x\bar{y}z + x\bar{y}\bar{z} + \bar{x}yz + \bar{x}\bar{y}z + \bar{x}\bar{y}\bar{z}$, and (c) $xyz + xy\bar{z} + x\bar{y}z + x\bar{y}\bar{z} + \bar{x}yz + \bar{x}\bar{y}z + \bar{x}\bar{y}\bar{z}$.

Solution: The Karnaugh maps for these sum-of-products expansions are shown in Figure 7. The grouping of blocks shows that minimal expansions into Boolean sums of Boolean products are (a) $x\bar{z} + \bar{y}\bar{z} + \bar{x}yz$, (b) $\bar{y} + \bar{x}z$, and (c) $x + \bar{y} + z$. ■

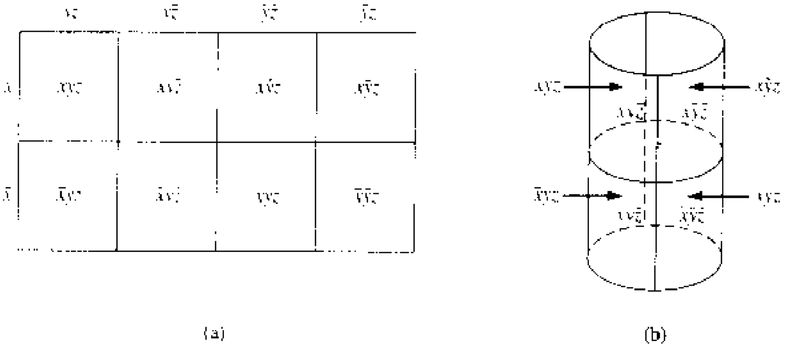
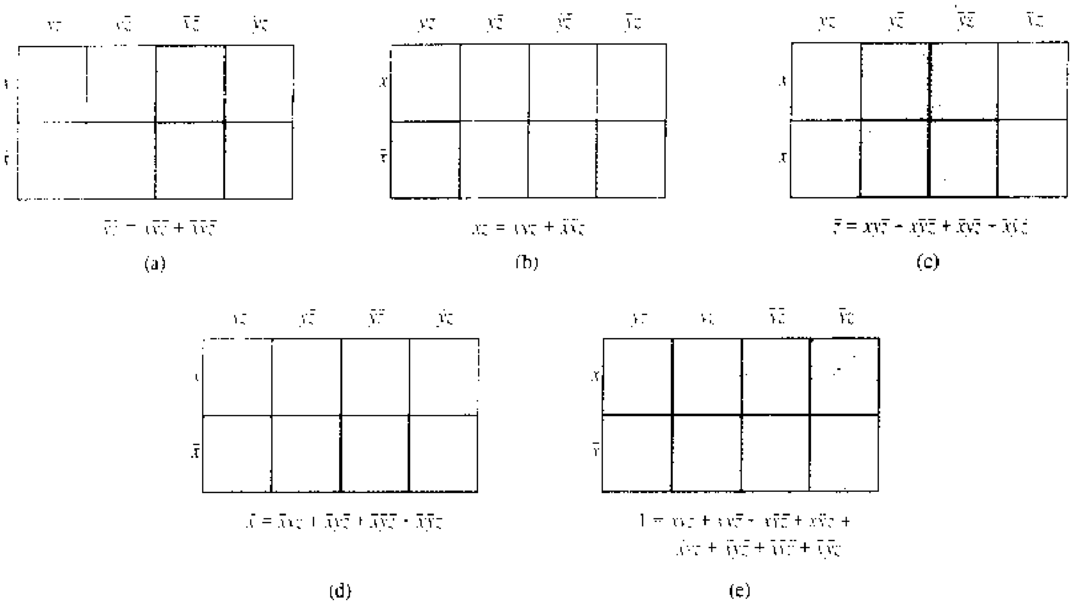


FIGURE 5 **Karnaugh Maps in Three Variables.**



yz $y\bar z$ $\bar y\bar z$ $\bar yz$
x
$\bar x$
$xz = xyz + x\bar yz$
$\bar z = xy\bar z + x\bar y\bar z + \bar xy\bar z + \bar x\bar y\bar z$
$\bar x = \bar xyz + \bar xy\bar z + \bar x\bar y\bar z + \bar x\bar yz$
$1 = xyz + xy\bar z + x\bar y\bar z + x\bar yz + \bar xyz + \bar xy\bar z + \bar x\bar y\bar z + \bar x\bar yz$
(a)
(b)
(c)
(d)
(e)

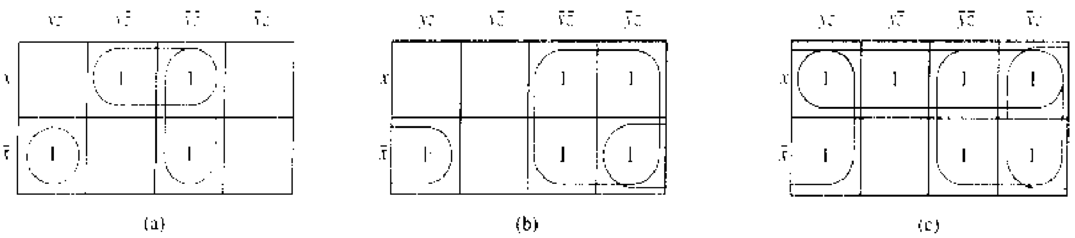
FIGURE 6 **Blocks in Karnaugh Maps in Three Variables.**

A Karnaugh map in four variables is a square that is divided into 16 squares. The squares represent the 16 possible minterms in four variables. One of the ways to form a Karnaugh map in four variables is shown in Figure 8.

Two squares are adjacent if and only if the minterms they represent differ in one literal. Consequently, each square is adjacent to four other squares. The Karnaugh map of a sum-of-products expansion in four variables can be thought of as lying on a torus, so that adjacent squares have a common boundary (see Exercise 20). The simplification of a sum-of-products expansion in four variables is carried out by identifying those blocks of 2, 4, 8, or 16 squares that represent minterms that can be combined. Each square representing a minterm must either be used to form a product using fewer literals, or be included in the expansion. In Figure 9 some examples of blocks that represent products of three literals, products of two literals, and a single literal are illustrated.

As is the case in Karnaugh maps in two and three variables, the goal is to identify the largest blocks of 1s in the map and to cover all the 1s using the fewest blocks needed, using the largest blocks first. The largest possible blocks are always used. The following example illustrates how Karnaugh maps in four variables are used.

EXAMPLE 4 Use Karnaugh maps to simplify the sum-of-products expansions (a) $wxyz + wxy\bar z + wx\bar y\bar z + w\bar xyz + w\bar x\bar yz + w\bar x\bar y\bar z + \bar wx\bar yz + \bar w\bar xyz + \bar w\bar xy\bar z$, (b) $wx\bar y\bar z + w\bar xyz + w\bar xy\bar z +$



yz $y\bar z$ $\bar y\bar z$ $\bar yz$
x
$\bar x$
1
(a)
(b)
(c)

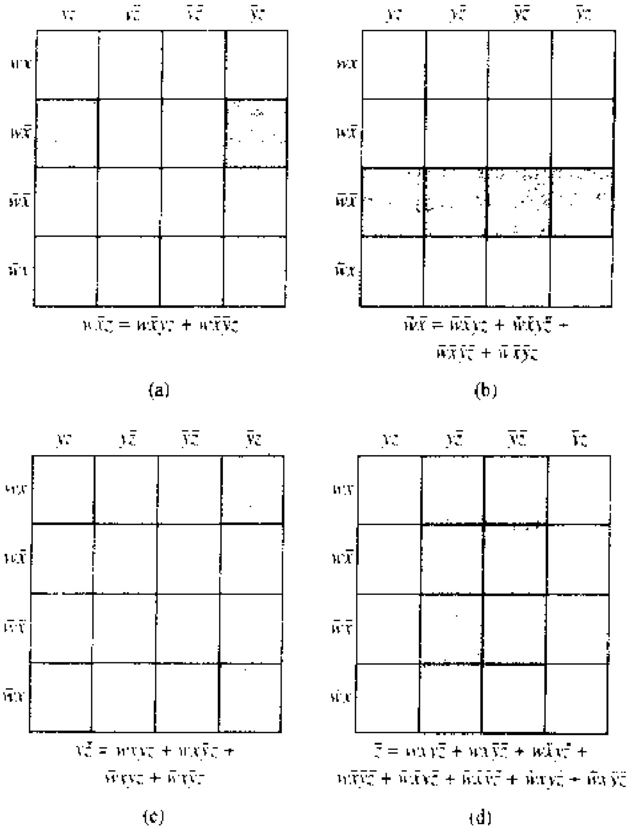
FIGURE 7 **Using Karnaugh Maps in Three Variables.**

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
wx	$wxyz$	$wxy\bar{z}$	$wx\bar{y}\bar{z}$	$wx\bar{y}z$
$w\bar{x}$	$w\bar{x}yz$	$w\bar{x}y\bar{z}$	$w\bar{x}\bar{y}\bar{z}$	$w\bar{x}\bar{y}z$
$\bar{w}\bar{x}$	$\bar{w}\bar{x}yz$	$\bar{w}\bar{x}y\bar{z}$	$\bar{w}\bar{x}\bar{y}\bar{z}$	$\bar{w}\bar{x}\bar{y}z$
$\bar{w}x$	$\bar{w}xyz$	$\bar{w}xy\bar{z}$	$\bar{w}x\bar{y}\bar{z}$	$\bar{w}x\bar{y}z$

FIGURE 8 Karnaugh Map in Four Variables.

$w\bar{x}\bar{y}\bar{z} + \bar{w}x\bar{y}\bar{z} + \bar{w}\bar{x}y\bar{z} + \bar{w}\bar{x}\bar{y}\bar{z}$, and (c) $wxy\bar{z} + wx\bar{y}\bar{z} + w\bar{x}yz + w\bar{x}y\bar{z} + w\bar{x}\bar{y}\bar{z} + \bar{w}xyz + \bar{w}xy\bar{z} + \bar{w}x\bar{y}\bar{z} + \bar{w}x\bar{y}z + \bar{w}\bar{x}y\bar{z} + \bar{w}\bar{x}\bar{y}\bar{z}$.

Solution: The Karnaugh maps for these expansions are shown in Figure 10. Using the blocks shown leads to the sum of products (a) $wyz + wx\bar{z} + w\bar{x}\bar{y} + \bar{w}\bar{x}y + \bar{w}x\bar{y}z$, (b) $\bar{y}\bar{z} + w\bar{x}y + \bar{x}y\bar{z}$, and (c) $\bar{z} + \bar{w}x + w\bar{x}y$. The reader should determine whether



$w\bar{x}z = w\bar{x}yz + w\bar{x}\bar{y}z$

(a)

$\bar{w}\bar{x} = \bar{w}\bar{x}yz + \bar{w}\bar{x}y\bar{z} + \bar{w}\bar{x}\bar{y}\bar{z} + \bar{w}\bar{x}\bar{y}z$

(b)

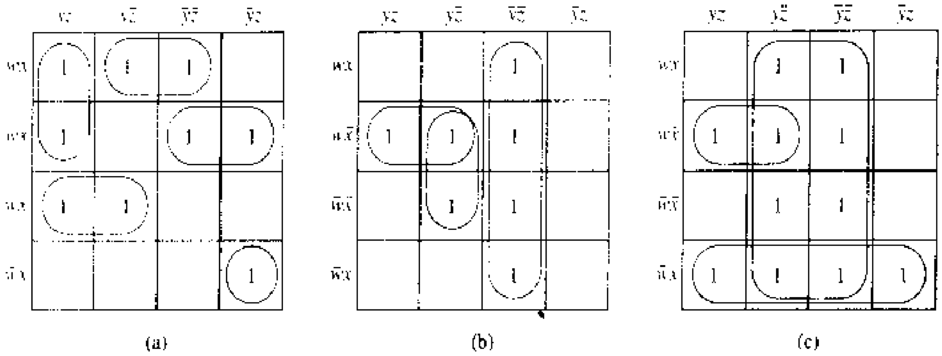
$x\bar{z} = wxy\bar{z} + wx\bar{y}\bar{z} + \bar{w}xy\bar{z} + \bar{w}x\bar{y}\bar{z}$

(c)

$\bar{z} = wxy\bar{z} + wx\bar{y}\bar{z} + w\bar{x}y\bar{z} + w\bar{x}\bar{y}\bar{z} + \bar{w}\bar{x}y\bar{z} + \bar{w}\bar{x}\bar{y}\bar{z} + \bar{w}xy\bar{z} + \bar{w}x\bar{y}\bar{z}$

(d)

FIGURE 9 Blocks in Karnaugh Maps in Four Variables.



(a) (b) (c)

FIGURE 10 Using Karnaugh Maps in Four Variables.

there are other choices of blocks in each part that lead to different sums of products representing these Boolean functions. ■

DON'T CARE CONDITIONS

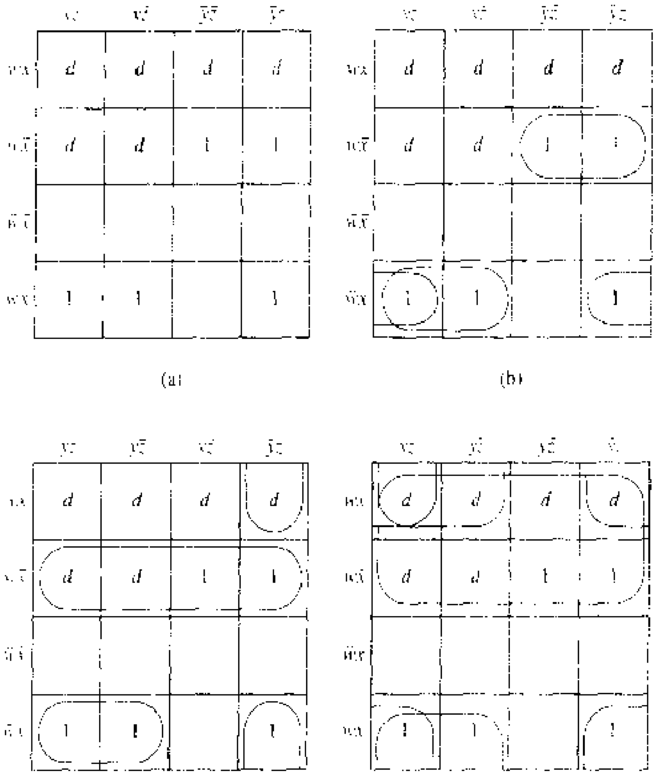
In some circuits we care only about the output for some combinations of input values, since the other combinations of input values never arise. This gives us freedom in producing a simple circuit with the desired output since the output values for all those combinations that never occur can be arbitrarily chosen. The values of the function for these combinations are called ***don't care* conditions.** A d is used in a Karnaugh map to mark those combinations of values of the variables for which the function can be arbitrarily assigned. In the simplification process we can assign 1s as values to those combinations of the input values that lead to the largest blocks in the Karnaugh map. This is illustrated in the following example.

EXAMPLE 5 One way to code decimal expansions using bits is to use the four bits of the binary expansion of each digit in the decimal expansion. For instance, 873 is encoded as 100001110011. This encoding of a decimal expansion is called a **binary coded decimal expansion**. Since there are 16 blocks of four bits and only 10 decimal digits, there are six combinations of four bits that are not used to encode digits. Suppose that a circuit is to be built that produces an output of 1 if the decimal digit is 5 or greater and an output of 0 if the decimal digit is less than 5. How can this circuit be simply built using *OR* gates, *AND* gates, and inverters?

Solution: Let $F(w, x, y, z)$ denote the output of the circuit, where $wxyz$ is a binary expansion of a decimal digit. The values of F are shown in Table 1. The Karnaugh map for F, with ds in the *don't care* positions, is shown in Figure 11(a). We can either include or exclude squares with ds from blocks. This gives us many possible choices for the blocks. For example, excluding all squares with ds and forming blocks as shown in Figure 11(b) produces the expression $wxy + wxy + wxz$. Including some of the ds and excluding others and forming blocks as shown in Figure 11(c) produces the expression $w\bar{x} + \bar{w}xy + x\bar{y}z$. Finally, including all the ds and using the blocks shown in Figure 11(d) produces the simplest expansion possible, namely, $F(x, y, z) = w + xy + xz$. ■

TABLE 1

Digit	*w*	*x*	*y*	*z*	*F*
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1


yz
$y\bar{z}$
$\bar{y}\bar{z}$
$\bar{y}z$
wx
$w\bar{x}$
$\bar{w}\bar{x}$
$\bar{w}x$
d
1

(a) (b)

(c) (d)

FIGURE 11 **The Karnaugh Map for *F* Showing Its *Don't Care* Positions.**

THE QUINE–McCLUSKEY METHOD

We have seen that Karnaugh maps can be used to produce minimal expansions of Boolean functions as Boolean sums of Boolean products. However, Karnaugh maps are awkward to use when there are more than four variables. Furthermore, the use of Karnaugh maps relies on visual inspection to identify terms to group. For these reasons there is a need for a procedure for simplifying sum-of-products expansions that can be mechanized. The *Quine–McCluskey method* is such a procedure. It can be used for Boolean functions in any number of variables. It was developed in the 1950s by W. V. Quine and E. J. McCluskey, Jr. Basically, the Quine–McCluskey method consists of two parts. The first part finds those terms that are candidates for inclusion in a minimal expansion as a Boolean sum of Boolean products. The second part determines which of these terms to actually use. We will show how this procedure works using an example.

EXAMPLE 6 We will show how the Quine–McCluskey method can be used to find a minimal expansion equivalent to

$$xyz + xyz + \bar{x}yz + \bar{x}\,\bar{y}z + x\,\bar{y}\,\bar{z}.$$

We will represent the minterms in this expansion by bit strings. The first bit will be 1 if x occurs and 0 if $\bar{x}$ occurs. The second bit will be 1 if y occurs and 0 if $\bar{y}$ occurs. The third bit will be 1 if z occurs and 0 if $\bar{z}$ occurs. We then group these terms according to the number of 1s in the corresponding bit strings. This information is shown in Table 2.

Minterms that can be combined are those that differ in exactly one literal. Hence, two terms that can be combined differ by exactly one in the number of 1s in the bit strings that represent them. When two minterms are combined into a product, this product contains two literals. A product in two literals is represented using a dash to denote

---

Edward J. McCluskey (born 1929). Edward McCluskey attended Bowdoin College and M.I.T., where he received his doctorate in Electrical Engineering in 1956. He joined Bell Telephone Laboratories in 1955, remaining there until 1959. McCluskey was professor of Electrical Engineering at Princeton University from 1959 until 1966, also serving as Director of the Computer Center at Princeton from 1961 to 1966. In 1967 he took a position as professor of Computer Science and Electrical Engineering at Stanford University, where he also served as director of the Digital Systems Laboratory from 1969 to 1978. McCluskey has worked in a variety of areas in computer science, including fault-tolerant computing, computer architecture, testing, and logic design. He is currently director of the Center for Reliable Computing at Stanford University. McCluskey is also an ACM Fellow.

Willard Van Orman Quine (born 1908). Willard Quine, born in Akron, Ohio, attended Oberlin College and later Harvard University, where he received his Ph.D. in Philosophy in 1932. He became a Junior Fellow at Harvard in 1933 and was appointed to a position on the faculty there in 1936. He has remained at Harvard his entire professional life, except for World War II, when he worked for the U.S. Navy decrypting messages from German submarines. Quine has always been interested in algorithms, but not in hardware. He arrived at his discovery of what is now called the Quine–McCluskey method as a device for teaching mathematical logic, rather than as a method for simplifying switching circuits. Quine is one of the most famous philosophers in the world. He has made fundamental contributions to the theory of knowledge, mathematical logic and set theory, and the philosophies of logic and language. His books including *New Foundations of Mathematical Logic* published in 1937 and *Word and Object* published in 1960, have had profound impact. Quine retired from Harvard in 1978 but continues to commute from his home in Beacon Hill to his office there. He still uses the 1927 Remington typewriter on which he prepared his doctoral thesis. Long ago he had an operation performed on this machine to add a few special symbols, removing the second period, the second comma, and the question mark. When asked whether he missed the question mark, he replied, "Well, you see, I deal in certainties." There is even a word *quine*, defined in the *New Hacker's Dictionary* as a program that generates a copy of its own source code as its complete output. Producing the shortest possible quine in a given programming language is a popular puzzle for hackers.

TABLE 2

Minterm	*Bit String*	*Number of 1s*
xyz	111	3
$x\bar{y}z$	101	2
$\bar{x}yz$	011	2
$\bar{x}\bar{y}z$	001	1
$\bar{x}\bar{y}\bar{z}$	000	0

the variable that does not occur. For instance, the minterms $x\bar{y}z$ and $\bar{x}\bar{y}z$, represented by bit strings 101 and 001, can be combined into $\bar{y}z$, represented by the string –01. All pairs of minterms that can be combined and the product formed from these combinations are shown in Table 3.

Next, all pairs of products of two literals that can be combined are combined into one literal. Two such products can be combined if they contain literals for the same two variables, and literals for only one of the two variables differ. In terms of the strings representing the products, these strings must have a dash in the same position and must differ in exactly one of the other two slots. We can combine the products yz and $\bar{y}z$, represented by strings –11 and –01, into z, represented by the string – –1. We show all the combinations of terms that can be formed in this way in Table 3.

In Table 3 we also indicate which terms have been used to form products with fewer literals; these terms will not be needed in a minimal expansion. The next step is to identify a minimal set of products needed to represent the Boolean function. We begin with all those products that were not used to construct products with fewer literals. Next, we form Table 4, which has a row for each candidate product formed by combining original terms, and a column for each original term; and we put an X in a position if the original term in the sum-of-products expansion was used to form this candidate product. In this case, we say that the candidate product **covers** the original minterm. We need to include at least one product that covers each of the original minterms. Consequently, whenever there is only one X in a column in the table, the product corresponding to

TABLE 3

			Step 1			*Step 2*		
	Term	*Bit String*		*Term*	*String*		*Term*	*String*
1	xyz	111	(1,2)	xz	1–1	(1,2,3,4)	z	– –1
2	$x\bar{y}z$	101	(1,3)	yz	–11			
3	$\bar{x}yz$	011	(2,4)	$\bar{y}z$	–01			
4	$\bar{x}\bar{y}z$	001	(3,4)	$\bar{x}z$	0–1			
5	$\bar{x}\bar{y}\bar{z}$	000	(4,5)	$\bar{x}\bar{y}$	00–			

TABLE 4

	xyz	$x\bar{y}z$	$\bar{x}yz$	$\bar{x}\bar{y}z$	$\bar{x}\bar{y}\bar{z}$
z	X	X	X	X	
$\bar{x}\bar{y}$				X	X

the row this X is in must be used. From Table 4 we see that both z and $x\,y$ are needed. Hence, the final answer is $z + \bar{x}\,\bar{y}$. ■

As was illustrated in Example 6, the Quine–McCluskey method uses the following sequence of steps to simplify a sum-of-products expression.

1. Express each minterm in n variables by a bit string of length n with a 1 in the ith position if x_i occurs and a 0 in this position if x_i occurs.
2. Group the bit strings according to the number of 1s in them.
3. Determine all products in $n - 1$ variables that can be formed by taking the Boolean sum of minterms in the expansion. Minterms that can be combined are represented by bit strings that differ in exactly one position. Represent these products in $n - 1$ variables with strings that have a 1 in the ith position if x_i occurs in the product, a 0 in this position if $\bar{x}_i$ occurs, and a dash in this position if there is no literal involving x_i in the product.
4. Determine all products in $n - 2$ variables that can be formed by taking the Boolean sum of the products in $n - 1$ variables found in the previous step. Products in $n - 1$ variables that can be combined are represented by bit strings which have a dash in the same position and differ in exactly one position.
5. Continue combining Boolean products into products in fewer variables as long as possible.
6. Find all the Boolean products which arose that were not used to form a Boolean product in one fewer literal.
7. Find the smallest set of these Boolean products so that the sum of these products represents the Boolean function. This is done by forming a table showing which minterms are covered by which products. Every minterm must be covered by at least one product. (This is the most difficult part of the procedure. It can be mechanized using a backtracking procedure.)

A final example will illustrate how this procedure is used to simplify a sum-of-products expansion in four variables.

EXAMPLE 7 Use the Quine–McCluskey method to simplify the sum-of-products expansion $wxyz + wxyz + w\bar{x}y\bar{z} + \bar{w}xyz + \bar{w}x\bar{y}z + \bar{w}\,\bar{x}yz + \bar{w}\,\bar{x}\,\bar{y}z$.

Solution: We first represent the minterms by bit strings and then group these terms together according to the number of 1s in the bit strings. This is shown in Table 5. All the Boolean products that can be formed by taking Boolean sums of these products are shown in Table 6.

TABLE 5

Term	*Bit String*	*Number of 1s*
$wxy\bar{z}$	1110	3
$w\bar{x}yz$	1011	3
$\bar{w}xyz$	0111	3
$w\bar{x}y\bar{z}$	1010	2
$\bar{w}x\bar{y}z$	0101	2
$\bar{w}\bar{x}yz$	0011	2
$\bar{w}\bar{x}\bar{y}z$	0001	1

TABLE 6

		Step 1		*Step 2*	
Term	*Bit String*	*Term*	*String*	*Term*	*String*
1 $wxy\bar{z}$	1110	(1,4) $wy\bar{z}$	1–10	(3,5,6,7) $\bar{w}z$	0– –1
2 $w\bar{x}yz$	1011	(2,4) $w\bar{x}y$	101–		
3 $\bar{w}xyz$	0111	(2,6) $\bar{x}yz$	–011		
4 $w\bar{x}y\bar{z}$	1010	(3,5) $\bar{w}xz$	01–1		
5 $\bar{w}x\bar{y}z$	0101	(3,6) $\bar{w}yz$	0–11		
6 $\bar{w}\bar{x}yz$	0011	(5,7) $\bar{w}\bar{y}z$	0–01		
7 $\bar{w}\bar{x}\bar{y}z$	0001	(6,7) $\bar{w}\bar{x}z$	00–1		

The only products that were not used to form products in fewer variables are $\bar{w}z$, $wy\bar{z}$, $w\bar{x}y$, and $\bar{x}yz$. In Table 7 we show the minterms covered by each of these products. To cover these minterms we must include $\bar{w}z$ and $wy\bar{z}$, since these products are the only products that cover $\bar{w}xyz$ and $wxy\bar{z}$, respectively. Once these two products are included, we see that only one of the two products left is needed. Consequently, we can take either $\bar{w}z + wy\bar{z} + w\bar{x}y$ or $\bar{w}z + wy\bar{z} + \bar{x}yz$ as the final answer. ■

TABLE 7

	$wxy\bar{z}$	$w\bar{x}yz$	$\bar{w}xyz$	$w\bar{x}y\bar{z}$	$\bar{w}x\bar{y}z$	$\bar{w}\bar{x}yz$	$\bar{w}\bar{x}\bar{y}z$
$\bar{w}z$			X		X	X	X
$wy\bar{z}$	X			X			
$w\bar{x}y$		X		X			
$\bar{x}yz$		X				X	

Exercises

1. **a)** Draw a Karnaugh map for a function in two variables and put a 1 in the square representing $\bar{x}y$.

b) What are the minterms represented by squares adjacent to this square?

2. Find the sum-of-products expansions represented by each of the following Karnaugh maps.

a)

	y	$\bar{y}$
x	1	
$\bar{x}$	1	1

b)

	y	$\bar{y}$
x	1	1
$\bar{x}$		

c)

	y	$\bar{y}$
x	1	1
$\bar{x}$	1	1

3. Draw the Karnaugh maps of the following sum-of-products expansions in two variables.

a) $x\bar{y}$

b) $xy + \bar{x}\,\bar{y}$

c) $xy + x\bar{y} + xy + x\,y$

4. Use a Karnaugh map to find a minimal expansion as a Boolean sum of Boolean products of each of the following functions of the Boolean variables x and y.

a) $\bar{x}y + \bar{x}\,\bar{y}$

b) $xy + x\bar{y}$

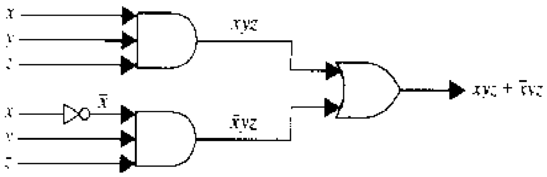
c) $xy + x\bar{y} + \bar{x}y + \bar{x}\,\bar{y}$

5. **a)** Draw a Karnaugh map for a function in three variables. Put a 1 in the square that represents $\bar{x}y\bar{z}$.

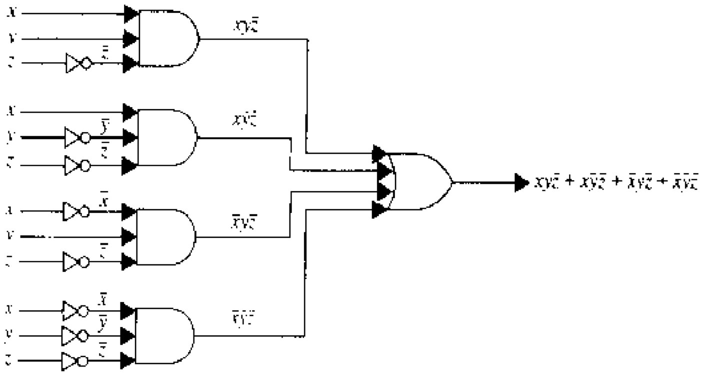
b) Which minterms are represented by squares adjacent to this square?

6. Use Karnaugh maps to find simpler circuits with the same output as each of the following circuits shown below.

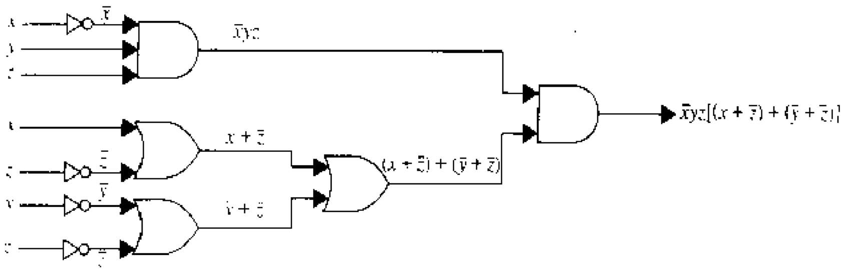
a)


x
y
z
x
$\bar{x}$
y
z
xyz
$\bar{x}yz$
$xyz + \bar{x}yz$

b)


x
y
z
$\bar{z}$
$xy\bar{z}$
x
y
$\bar{y}$
z
$\bar{z}$
$x\bar{y}\bar{z}$
x
$\bar{x}$
y
z
$\bar{z}$
$\bar{x}y\bar{z}$
x
$\bar{x}$
y
$\bar{y}$
z
$\bar{z}$
$\bar{x}\bar{y}\bar{z}$
$xy\bar{z} + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}\bar{z}$

c)


x
$\bar{x}$
y
z
$\bar{x}yz$
x
z
$\bar{z}$
$x + \bar{z}$
y
$\bar{y}$
z
$\bar{z}$
$\bar{y} + \bar{z}$
$(x + \bar{z}) + (\bar{y} + \bar{z})$
$\bar{x}yz[(x + \bar{z}) + (\bar{y} + \bar{z})]$

7. Draw the Karnaugh maps of the following sum-of-products expansions in three variables.
   a) $x\bar{y}\bar{z}$
   b) $xyz + \bar{x}yz$
   c) $xyz + xy\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}z$
8. Use a Karnaugh map to find a minimal expansion as a Boolean sum of Boolean products of each of the following functions in the variables x, y, and z.
   a) $\bar{x}yz + \bar{x}\bar{y}z$
   b) $xyz + xyz + xy\bar{z} + \bar{x}yz$
   c) $xyz + x\bar{y}z + x\bar{y}z + xyz + \bar{x}\bar{y}z$
   d) $xyz + x\bar{y}z + xy\bar{z} + \bar{x}yz + \bar{x}y\bar{z} + \bar{x}\bar{y}z$
9. a) Draw a Karnaugh map for a function in four variables. Put a 1 in the square that represents $\bar{w}xy\bar{z}$.
   b) Which minterms are represented by squares adjacent to this square?
10. Use a Karnaugh map to find a minimal expansion as a Boolean sum of Boolean products of each of the following functions in the variables w, x, y, and z.
    a) $wxyz + wxyz + wx\bar{y}\bar{z} + wxy\bar{z} + w\bar{x}yz$
    b) $wxyz + wx\bar{y}z + wxyz + \bar{w}x\bar{y}z + \bar{w}\bar{x}y\bar{z} + \bar{w}\bar{x}\bar{y}z$
    c) $wxyz + wxy\bar{z} + wx\bar{y}z + w\bar{x}\bar{y}z + w\bar{x}\bar{y}\bar{z} + \bar{w}x\bar{y}z + wxyz + \bar{w}x\bar{y}z$
    d) $wxyz + wxy\bar{z} + wx\bar{y}z + wxyz + w\bar{x}y\bar{z} + \bar{w}xyz + \bar{w}xyz + \bar{w}\bar{x}yz + \bar{w}\bar{x}\bar{y}z$
11. a) How many squares does a Karnaugh map in five variables have?
    b) How many squares are adjacent to a given square in a Karnaugh map in five variables?

*12. Use Karnaugh maps to find a minimal expansion as a Boolean sum of Boolean products of Boolean functions that have as input the binary code for each decimal digit and produce as output a 1 if and only if the digit corresponding to the input is
   a) odd.
   b) not divisible by 3.
   c) not 4, 5, or 6.

*13. Suppose that there are five members on a committee, but that Smith and Jones always vote the opposite of Marcus. Design a circuit that implements majority voting of the committee using this relationship between votes.

14. Use the Quine–McCluskey method to simplify the sum-of-products expansions in Example 3.
15. Use the Quine–McCluskey method to simplify the sum-of-products expansions in Exercise 8.
16. Use the Quine–McCluskey method to simplify the sum-of-products expansions in Example 4.
17. Use the Quine–McCluskey method to simplify the sum-of-products expansions in Exercise 10.

*18. Explain how Karnaugh maps can be used to simplify product-of-sums expansions in three variables. (*Hint:* Mark with a 0 all the maxterms in an expansion and combine blocks of maxterms.)

19. Use the method from Exercise 18 to simplify the product-of-sums expansion $(x + y + z)(x + y + \bar{z})(x + \bar{y} + \bar{z})(x + \bar{y} + \bar{z})(\bar{x} + y + z)$.

*20. Draw a Karnaugh map for the 16 minterms in four Boolean variables on the surface of a torus.

21. Build a circuit using *OR* gates, *AND* gates, and inverters that produces an output of 1 if a decimal digit, encoded using a binary coded decimal expansion, is divisible by 3, and an output of 0 otherwise.

In Exercises 22–24 find a minimal sum-of-products expansion, given the Karnaugh map shown with *don't care* conditions indicated with *d*s.

22.

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
wx	d	1	d	1
$w\bar{x}$		d	d	
$\bar{w}\bar{x}$		d	1	
$\bar{w}x$		1	d	

23.

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
wx	1			1
$w\bar{x}$		d	1	
$\bar{w}\bar{x}$		1	d	
$\bar{w}x$	d			d

24.

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
wx		d	d	1
$w\bar{x}$	d	d	1	d
$\bar{w}\bar{x}$				
$\bar{w}x$	1	1	1	d

Key Terms and Results

TERMS

Boolean variable: a variable that assumes only the values 0 and 1

$\bar{x}$ (complement of x): an expression with the value 1 when x has the value 0 and the value 0 when x has the value 1

$x \cdot y$ (or xy) (Boolean product or conjunction of x and y): an expression with the value 1 when both x and y have the value 1 and the value 0 otherwise

$x + y$ (Boolean sum or disjunction of x and y): an expression with the value 1 when either x or y, or both, has the value 1, and 0 otherwise

Boolean expressions: the expressions obtained recursively by specifying that $0, 1, x_1, \ldots, x_n$ are Boolean expressions and $\bar{E}_1$, $(E_1 + E_2)$, and (E_1E_2) are Boolean expressions if E_1 and E_2 are

dual of a Boolean expression: the expression obtained by interchanging + signs and · signs and interchanging 0s and 1s

Boolean function of degree n: a function from B^n to B where $B = \{0, 1\}$

Boolean algebra: a set B with two binary operations $\vee$ and $\wedge$, elements 0 and 1, and a complementation operator $^{-}$ that satisfies the identity, domination, associative, commutative, and distributive laws

literal of the Boolean variable x: either x or $\bar{x}$

***minterm* of $x_1, x_2, \ldots, x_n$:** a Boolean product $y_1 y_2 \cdots y_n$ where each y_i is either x_i or $\bar{x}_i$

sum-of-products expansion (or **disjunctive normal form**): the representation of a Boolean function as a disjunction of minterms

functionally complete: a set of Boolean operators is called functionally complete if every Boolean function can be represented using these operators

$x \mid y$ (or x *NAND* y): the expression that has the value 0 when both x and y have the value 1 and the value 1 otherwise

$x \downarrow y$ (or x *NOR* y): the expression that has the value 0 when either x or y or both have the value 1 and the value 0 otherwise

inverter: a device that accepts the value of a Boolean variable as input and produces the complement of the input

***OR* gate:** a device that accepts the values of two or more Boolean variables as input and produces their Boolean sum as output

***AND* gate:** a device that accepts the values of two or more Boolean variables as input and produces their Boolean product as output

half adder: a circuit that adds two bits, producing a sum bit and a carry bit

full adder: a circuit that adds two bits and a carry, producing a sum bit and a carry bit

Karnaugh map for n variables: a rectangle divided into 2^n squares where each square represents a minterm in the variables

RESULTS

The identities for Boolean algebra (see Table 5 in Section 9.1).

An identity between Boolean functions represented by Boolean expressions remains valid when the duals of both sides of the identity are taken.

Every Boolean function can be represented by a sum-of-products expansion.

Each of the sets $\{+, ^{-}\}$ and $\{\cdot, ^{-}\}$ is functionally complete.

Each of the sets $\{\downarrow\}$ and $\{\mid\}$ is functionally complete.

The use of Karnaugh maps to minimize Boolean expressions.

The Quine–McCluskey method for minimizing Boolean expressions.

Review Questions

1. Define a Boolean function of degree n.
2. How many Boolean functions of degree 2 are there?
3. Give a recursive definition of the set of Boolean expressions.
4. a) What is the dual of a Boolean expression?
   b) What is the duality principle? How can it be used to find new identities involving Boolean expressions?
5. Explain how to construct the sum-of-products expansion of a Boolean function.
6. a) What does it mean for a set of operators to be functionally complete?
   b) Is the set $\{+, \cdot\}$ functionally complete?
   c) Are there sets of a single operator that are functionally complete?
7. Explain how to build a circuit for a light controlled by two switches using *OR* gates, *AND* gates, and inverters.
8. Construct a half adder using *OR* gates, *AND* gates, and inverters.

9. Is there a single type of logic gate that can be used to build all circuits that can be built using *OR* gates, *AND* gates, and inverters?

10. **a)** Explain how Karnaugh maps can be used to simplify sum-of-products expansions in three Boolean variables.

b) Use a Karnaugh map to simplify the sum-of-products expansion $xyz + x\bar{y}z + xy\,z + xyz + x\,y\,z$.

11. **a)** Explain how Karnaugh maps can be used to simplify sum-of-products expansions in four Boolean variables.

b) Use a Karnaugh map to simplify the sum-of-products expansion $wxyz + wxyz + wx\bar{y}z + wx\bar{y}\,\bar{z} + w\bar{x}yz + w\bar{x}\,\bar{y}z + wxyz + w\bar{x}yz + \bar{w}\,xyz$.

12. **a)** What is a *don't care* condition?

b) Explain how *don't care* conditions can be used to build a circuit using *OR* gates, *AND* gates, and inverters that produces an output of 1 if a decimal digit is 6 or greater, and an output of 0 if this digit is less than 6.

13. **a)** Explain how to use the Quine–McCluskey method to simplify sum-of-products expansions.

b) Use this method to simplify $xy\bar{z} + xyz + \bar{x}y\bar{z} + x\bar{y}z + \bar{x}\,y\,z$.

Supplementary Exercises

1. For which values of the Boolean variables x, y, and z does

a) $x + y + z = xyz$? **b)** $x(y + z) = x + yz$?

c) $\bar{x}\,y\,z = x + y + z$?

2. Let x and y belong to $\{0, 1\}$. Does it necessarily follow that $x = y$ if there exists a value z in $\{0, 1\}$ such that

a) $xz = yz$? **b)** $x + z = y + z$?

c) $x \oplus z = y \oplus z$? **d)** $x \downarrow z = y \downarrow z$?

e) $x \mid z = y \mid z$?

A Boolean function F is called **self-dual** if and only if $F(x_1, \ldots, x_n) = \overline{F}(\bar{x}_1, \ldots, \bar{x}_n)$.

3. Which of the following functions are self-dual?

a) $F(x, y) = x$ **b)** $F(x, y) = xy + \bar{x}\,\bar{y}$

c) $F(x, y) = x + y$ **d)** $F(x, y) = xy + \bar{x}y$

4. Give an example of a self-dual Boolean function of three variables.

***5.** How many Boolean functions of degree n are self-dual?

We define the relation $\leq$ on the set of Boolean functions of degree n so that $F \leq G$ where F and G are Boolean functions if and only if $G(x_1, x_2, \ldots, x_n) = 1$ whenever $F(x_1, x_2, \ldots, x_n) = 1$.

6. Determine whether $F \leq G$ or $G \leq F$ for the following pairs of functions.

a) $F(x, y) = x$, $G(x, y) = x + y$

b) $F(x, y) = x + y$, $G(x, y) = xy$

c) $F(x, y) = x$, $G(x, y) = x + y$

7. Show that if F and G are Boolean functions of degree n, then

a) $F \leq F + G$. **b)** $FG \leq F$.

8. Show that if F, G, and H are Boolean functions of degree n, then $F + G \leq H$ if and only if $F \leq H$ and $G \leq H$.

***9.** Show that the relation $\leq$ is a partial ordering on the set of Boolean functions of degree n.

***10.** Draw the Hasse diagram for the poset consisting of the set of the 16 Boolean functions of degree 2 (shown in Table 3 of Section 9.1) with the partial ordering $\leq$.

***11.** For each of the following equalities either prove it is an identity or find a set of values of the variables for which it does not hold.

a) $x \mid (y \mid z) = (x \mid y) \mid z$

b) $x \downarrow (y \downarrow z) = (x \downarrow y) \downarrow (x \downarrow z)$

c) $x \downarrow (y \mid z) = (x \downarrow y) \mid (x \downarrow z)$

Define the Boolean operator $\odot$ as follows: $1 \odot 1 = 1$, $1 \odot 0 = 0$, $0 \odot 1 = 0$, and $0 \odot 0 = 1$.

12. Show that $x \odot y = xy + \bar{x}\,\bar{y}$.

13. Show that $x \odot y = \overline{(x \oplus y)}$.

14. Show that each of the following identities holds.

a) $x \odot x = 1$ **b)** $x \odot \bar{x} = 0$

c) $x \odot y = y \odot x$

15. Is it always true that $(x \odot y) \odot z = x \odot (y \odot z)$?

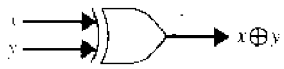
***16.** Determine whether the set $\{\odot\}$ is functionally complete.

***17.** How many of the 16 Boolean functions in two variables x and y can be represented using only the following set of operators, variables x and y, and values 0 and 1?

a) $\{\ \}$ **b)** $\{\cdot\}$

c) $\{+\}$ **d)** $\{\cdot, +\}$

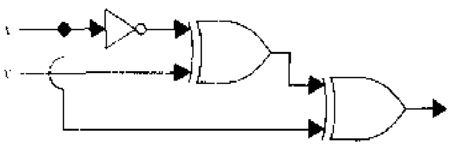
The notation for an *XOR* gate, which produces the output $x \oplus y$ from x and y, is as follows:



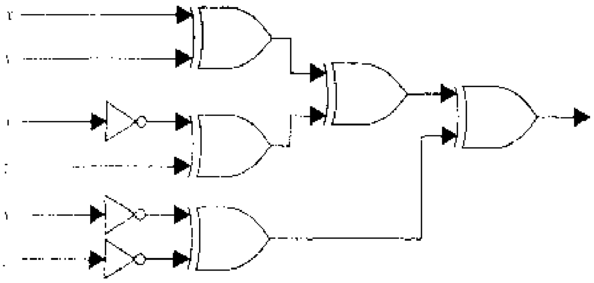
18. Determine the output of circuits (a) and (b) shown at the top of the facing page.

19. Show how a half adder can be constructed using fewer gates than are used in Figure 8 of Section 9.3 when *XOR* gates can be used in addition to *OR* gates, *AND* gates, and inverters.

a)



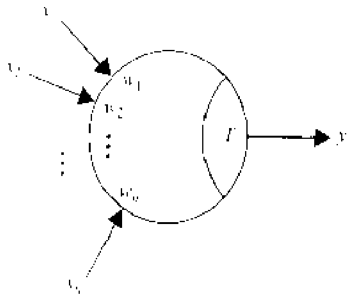
b)



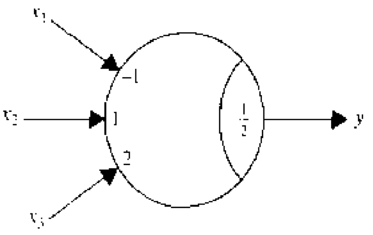
---

20. Design a circuit that determines whether three or more of four individuals on a committee vote yes on an issue, where each individual uses a switch for the voting.

A **threshold gate** produces an output y that is either 0 or 1 given a set of input values for the Boolean variables $x_1, x_2, \ldots, x_n$. A threshold gate has a **threshold value** T, which is a real number, and **weights** $w_1, w_2, \ldots, w_n$, each of which is a real number. The output y of the threshold gate is 1 if and only if $w_1x_1 + w_2x_2 + \cdots + w_nx_n \geq T$. The threshold gate with threshold value T and weights $w_1, w_2, \ldots, w_n$ is represented by the following diagram. Threshold gates are useful in modeling in neurophysiology and in artificial intelligence.



21. A threshold gate represents a Boolean function. Find a Boolean expression for the Boolean function represented by the following threshold gate.



22. A Boolean function that can be represented by a threshold gate is called a **threshold function.** Show that each of the following functions is a threshold function.

a) $F(x) = \bar{x}$
b) $F(x, y) = x + y$
c) $F(x, y) = xy$
d) $F(x, y) = x \mid y$
e) $F(x, y) = x \downarrow y$
f) $F(x, y, z) = x + yz$
g) $F(w, x, y, z) = w + xy + z$
h) $F(w, x, y, z) = wxz + x\bar{y}z$

***23.** Show that $F(x, y) = x \oplus y$ is not a threshold function.

***24.** Show that $F(w, x, y, z) = wx + yz$ is not a threshold function.

Computer Projects

WRITE PROGRAMS WITH THE FOLLOWING INPUT AND OUTPUT.

1. Given the values of two Boolean variables x and y, find the values of $x + y$, xy, $x \oplus y$, $x \mid y$, and $x \downarrow y$.

2. Construct a table listing the set of values of all 256 Boolean functions of degree 3.

3. Given the values of a Boolean function in n variables, where n is a positive integer, construct the sum-of-products expansion of this function.

4. Given the table of values of a Boolean function, express this function using only the operators $\cdot$ and $\bar{\ }$.

5. Given the table of values of a Boolean function, express this function using only the operators $+$ and $\bar{\ }$.

***6.** Given the table of values of a Boolean function, express this function using only the operator $\mid$.

*7. Given the table of values of a Boolean function, express this function using only the operator ↓.

8. Given the table of values of a Boolean function of degree 3, construct its Karnaugh map.

9. Given the table of values of a Boolean function of degree 4, construct its Karnaugh map.

**10. Given the table of values of a Boolean function, use the Quine–McCluskey method to find a minimal sum-of-products representation of this function.

11. Given a threshold value and a set of weights for a threshold gate and the values of the n Boolean variables in the input, determine the output of this gate.

12. Given a positive integer n, construct a random Boolean expression in n variables in disjunctive normal form.

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. Compute the number of Boolean functions of degrees 7, 8, 9, and 10.
2. Construct a table of the Boolean functions of degree 3.
3. Construct a table of the Boolean functions of degree 4.
4. Express each of the different Boolean expressions in three variables in disjunctive normal form with just the *NAND* operator, using as few *NAND* operators as possible. What is the largest number of *NAND* operators required?
5. Express each of the different Boolean expressions in disjunctive normal form in four variables using just the *NOR* operator, with as few *NOR* operators as possible. What is the largest number of *NOR* operators required?
6. Randomly generate 10 different Boolean expressions in four variables and determine the average number of steps required to minimize them using the Quine–McCluskey method.
7. Randomly generate 10 different Boolean expressions in five variables and determine the average number of steps required to minimize them using the Quine–McCluskey method.

Writing Projects

RESPOND TO THE FOLLOWING QUESTIONS WITH ESSAYS USING OUTSIDE SOURCES.

1. Describe some of the early machines devised to solve problems in logic, such as the Stanhope Demonstrator, Jevons's Logic Machine, and the Marquand Machine.
2. Explain the difference between combinational circuits and sequential circuits. Then explain how *flip-flops* are used to build sequential circuits.
3. Define a *shift register* and discuss how shift registers are used. Show how to build shift registers using flip-flops and logic gates.
4. Show how *multipliers* can be built using logic gates.
5. Find out how logic gates are physically constructed. Discuss whether *NAND* and *NOR* gates are used in building circuits.
6. Explain how *dependency notation* can be used to describe complicated switching circuits.
7. Describe how multiplexers are used to build switching circuits.
8. Explain the advantages of using threshold gates to construct switching circuits. Illustrate this by using threshold gates to construct half and full adders.
9. Describe the concept of *hazard-free switching circuits* and give some of the principles used in designing such circuits.
10. Explain how to use Karnaugh maps to minimize functions of five or six variables.
11. Describe what is meant by the *functional decomposition* of a Boolean function of n variables and discuss procedures for decomposing Boolean functions into a composition of Boolean functions with fewer variables.

Modeling Computation

10

Computers can perform many tasks. Given a task, two questions arise. The first is: Can it be carried out using a computer? Once we know that this first question has an affirmative answer, we can ask the second question: How can the task be carried out? Models of computation are used to help answer these questions.

We will study three types of structures used in models of computation, namely, grammars, finite-state machines, and Turing machines. Grammars are used to generate the words of a language and to determine whether a word is in a language. Formal languages, which are generated by grammars, provide models for both natural languages, such as English, and for programming languages, such as Pascal, Fortran, Prolog, and C. In particular, grammars are extremely important in the construction and theory of compilers. The grammars that we will discuss were first used by the American linguist Noam Chomsky in the 1950s.

Various types of finite-state machines are used in modeling. All finite-state machines have a set of states, including a starting state, an input alphabet, and a transition function that assigns a next state to every pair of a state and an input. The states of a finite-state machine give it limited memory capabilities. Some finite-state machines produce an output symbol for each transition; these machines can be used to model many kinds of machines, including vending machines, delay machines, binary adders, and language recognizers. We will also study finite-state machines that have no output but do have final states. Such machines are extensively used in language recognition. The strings that are recognized are those which take the starting state to a final state. The concepts of grammars and finite-state machines can be tied together. We will characterize those sets that are recognized by a finite-state machine and show that these are precisely the sets which are generated by a certain type of grammar.

Finally, we will introduce the concept of a Turing machine. We will show how Turing machines can be used to recognize sets. We will also show how Turing machines can be used to compute number-theoretic functions. Finally, we will discuss the Church–Turing thesis, which states that every effective computation can be carried out using a Turing machine.

10.1

Languages and Grammars

INTRODUCTION

Words in the English language can be combined in various ways. The grammar of English tells us whether a combination of words is a valid sentence. For instance, *the frog writes neatly* is a valid sentence, since it is formed from a noun phrase, *the frog*, made up of the article *the* and the noun *frog*, followed by a verb phrase, *writes neatly*, made up of the verb *writes* and the adverb *neatly*. We do not care that this is a nonsensical statement, because we are concerned only with the **syntax,** or form, of the sentence, and not its **semantics,** or meaning. We also note that the combination of words *swims quickly mathematics* is not a valid sentence because it does not follow the rules of English grammar.

The syntax of a **natural language,** that is, a spoken language, such as English, French, German, or Spanish, is extremely complicated. In fact, it does not seem possible to specify all the rules of syntax for a natural language. Research in the automatic translation of one language to another has led to the concept of a **formal language,** which, unlike a natural language, is specified by a well-defined set of rules of syntax. Rules of syntax are important not only in linguistics, the study of natural languages, but also in the study of programming languages.

We will describe the sentences of a formal language using a grammar. The use of grammars helps when we consider the two classes of problems that arise most frequently in applications to programming languages: (1) How can we determine whether a combination of words is a valid sentence in a formal language? (2) How can we generate the valid sentences of a formal language?

Before giving a technical definition of a grammar, we will describe an example of a grammar that generates a subset of English. This subset of English is defined using a list of rules that describe how a valid sentence can be produced. We specify that

1. a **sentence** is made up of a **noun phrase** followed by a **verb phrase;**
2. a **noun phrase** is made up of an **article** followed by an **adjective** followed by a **noun,** or
3. a **noun phrase** is made up of an **article** followed by a **noun;**
4. a **verb phrase** is made up of a **verb** followed by an **adverb,** or
5. a **verb phrase** is made up of a **verb;**
6. an article is *a,* or
7. an article is *the;*
8. an adjective is *large,* or
9. an adjective is *hungry;*
10. a noun is *rabbit,* or
11. a noun is *mathematician;*
12. a verb is *eats,* or
13. a verb is *hops;*
14. an adverb is *quickly,* or
15. an adverb is *wildly.*

From these rules we can form valid sentences using a series of replacements until no more rules can be used. For instance, we can follow the sequence of replacements:

sentence
noun phrase **verb phrase**
article **adjective** **noun** **verb phrase**
article **adjective** **noun** **verb** **adverb**
the **adjective** **noun** **verb** **adverb**
the large **noun** **verb** **adverb**
the large rabbit **verb** **adverb**

the large rabbit hops **adverb**

the large rabbit hops quickly

to obtain a valid sentence. It is also easy to see that some other valid sentences are: *a hungry mathematician eats wildly, a large mathematician hops, the rabbit eats quickly,* and so on. Also, we can see that *the quickly eats mathematician* is not a valid sentence.

PHRASE-STRUCTURE GRAMMARS

Before we give a formal definition of a grammar, we introduce a little terminology.

DEFINITION 1. **A *vocabulary* (or *alphabet*) V is a finite, nonempty set of elements called *symbols*. A *word* (or *sentence*) over V is a string of finite length of elements of V. The *empty string* or *null string*, denoted by λ, is the string containing no symbols. The set of all words over V is denoted by V^*. A *language over* V is a subset of V^*.**

Note that λ, the empty string, is the string containing no symbols. It is different from $\emptyset$, the empty set. It follows that $\{\lambda\}$ is the set containing exactly one string, namely, the empty string.

Languages can be specified in various ways. One way is to list all the words in the language. Another is to give some criteria that a word must satisfy to be in the language. In this section we describe another important way to specify a language, namely, through the use of a grammar, such as the set of rules we gave in the introduction to this section. A grammar provides a set of symbols of various types and a set of rules for producing words. More precisely, a grammar has a **vocabulary** V, which is a set of symbols used to derive members of the language. Some of the elements of the vocabulary cannot be replaced by other symbols. These are called **terminals**, and the other members of the vocabulary, which can be replaced by other symbols, are called **nonterminals.** The sets of terminals and nonterminals are usually denoted by T and N, respectively. In the example given in the introduction of the section, the set of terminals is {*a, the, rabbit, mathematician, hops, eats, quickly, wildly*}, and the set of nonterminals is {**sentence, noun phrase, verb phrase, adjective, article, noun, verb, adverb**}. There is a special member of the vocabulary called the **start symbol**, denoted by S, which is the element of the vocabulary that we always begin with. In the example in the introduction, the start symbol is **sentence.** The rules that specify when we can replace a string from V^*, the set of all strings of elements in the vocabulary, with another string are called the **productions** of the grammar. We denote by $w_0 \rightarrow w_1$ the production which specifies that w_0 can be replaced by w_1. The productions in the grammar given in the introduction of this section were listed. The first production, written using this notation, is **sentence** → **noun phrase verb phrase**. We summarize with the following definition.

DEFINITION 2. **A *phrase-structure grammar* $G = (V, T, S, P)$ consists of a vocabulary V, a subset T of V consisting of terminal elements, a start symbol S from V, and a set of productions P. The set $V - T$ is denoted by N. Elements of N are called *nonterminal symbols*. Every production in P must contain at least one nonterminal on its left side.**

EXAMPLE 1 Let $G = \{V, T, S, P\}$ where $V = \{a, b, A, B, S\}$, $T = \{a, b\}$, S is the start symbol, and $P = \{S \to ABa, A \to BB, B \to ab, AB \to b\}$. G is an example of a phrase-structure grammar. ■

We will be interested in the words that can be generated by the productions of a phrase-structure grammar.

DEFINITION 3. Let $G = (V, T, S, P)$ be a phrase-structure grammar. Let $w_0 = lz_0r$ (that is, the concatenation of l, z_0, and r) and $w_1 = lz_1r$ be strings over V. If $z_0 \to z_1$ is a production of G, we say that w_1 is *directly derivable* from w_0 and we write $w_0 \Rightarrow w_1$. If $w_0, w_1, \ldots, w_n$, $n \geq 0$, are strings over V such that $w_0 \Rightarrow w_1, w_1 \Rightarrow w_2, \ldots, w_{n-1} \Rightarrow w_n$, then we say that w_n is *derivable from* w_0, and we write $w_0 \overset{*}{\Rightarrow} w_n$. The sequence of steps used to obtain w_n from w_0 is called a *derivation*.

EXAMPLE 2 The string $Aaba$ is directly derivable from ABa in the grammar in Example 1 since $B \to ab$ is a production in the grammar. The string $abababa$ is derivable from ABa since $ABa \Rightarrow Aaba \Rightarrow BBaba \Rightarrow Bababa \Rightarrow abababa$, using the productions $B \to ab$, $A \to BB$, $B \to ab$, and $B \to ab$ in succession. ■

DEFINITION 4. Let $G = \{V, T, S, P\}$ be a phrase-structure grammar. The *language generated by* G (or the *language of* G), denoted by $L(G)$, is the set of all strings of terminals that are derivable from the starting state S. In other words,

$$L(G) = \{w \in T^* \mid S \overset{*}{\Rightarrow} w\}.$$

In the following two examples we find the language generated by a phrase-structure grammar.

EXAMPLE 3 Let G be the grammar with vocabulary $V = \{S, A, a, b\}$, set of terminals $T = \{a, b\}$, starting symbol S, and productions $P = \{S \to aA, S \to b, A \to aa\}$. What is $L(G)$, the language of this grammar?

Solution: From the start state S we can derive aA using the production $S \to aA$. We can also use the production $S \to b$ to derive b. From aA the production $A \to aa$ can be used to derive aaa. No additional words can be derived. Hence $L(G) = \{b, aaa\}$. ■

EXAMPLE 4 Let G be the grammar with vocabulary $V = \{S, 0, 1\}$, set of terminals $T = \{0, 1\}$, starting symbol S, and productions $P = \{S \to 11S, S \to 0\}$. What is $L(G)$, the language of this grammar?

Solution: From S we can derive 0 using $S \to 0$, or $11S$ using $S \to 11S$. From $11S$ we can derive either 110 or $1111S$. From $1111S$ we can derive 11110 and $111111S$.

At any stage of a derivation we can either add two 1s at the end of the string or terminate the derivation by adding a 0 at the end of the string. We surmise that $L(G) = \{0, 110, 11110, 1111110, \ldots\}$, the set of all strings that begin with an even number of 1s and end with a 0. This can be proved using an inductive argument that shows that after n productions have been used, the only strings of terminals generated are those consisting of $n - 1$ or fewer concatenations of 11 followed by 0. (This is left as an exercise for the reader.) ◼

The problem of constructing a grammar that generates a given language often arises. The next three examples describe problems of this kind.

EXAMPLE 5 Give a phrase-structure grammar that generates the set $\{0^n 1^n \mid n = 0, 1, 2, \ldots\}$.

Solution: Two productions can be used to generate all strings consisting of a string of 0s followed by a string of the same number of 1s, including the null string. The first builds up successively longer strings in the language by adding a 0 at the start of the string and a 1 at the end. The second production replaces S with the empty string. The solution is the grammar $G = (V, T, S, P)$, where $V = \{0, 1, S\}$, $T = \{0, 1\}$, S is the starting symbol, and the productions are

$$S \rightarrow 0S1$$
$$S \rightarrow \lambda.$$

The verification that this grammar generates the correct set is left as an exercise for the reader. ◼

The last example involved the set of strings made up of 0s followed by 1s, where the number of 0s and 1s are the same. The next example considers the set of strings consisting of 0s followed by 1s, where the number of 0s and 1s may differ.

EXAMPLE 6 Find a phrase-structure grammar to generate the set $\{0^m 1^n \mid m \text{ and } n \text{ are nonnegative integers}\}$.

Solution: We will give two grammars G_1 and G_2 that generate this set. This will illustrate that two grammars can generate the same language.

The grammar G_1 has alphabet $V = \{S, 0, 1\}$, terminals $T = \{0, 1\}$, and productions $S \rightarrow 0S$, $S \rightarrow S1$, and $S \rightarrow \lambda$. G_1 generates the correct set, since using the first production m times puts m 0s at the beginning of the string, and using the second production n times puts n 1s at the end of the string. The details of this verification are left to the reader.

The grammar G_2 has alphabet $V = \{S, A, 0, 1\}$, terminals $T = \{0, 1\}$, and productions $S \rightarrow 0S$, $S \rightarrow 1A$, $S \rightarrow 1$, $A \rightarrow 1A$, $A \rightarrow 1$, $S \rightarrow \lambda$. The details that this grammar generates the correct set are left as an exercise for the reader. ◼

Sometimes a set that is easy to describe can be generated only by a complicated grammar. The next example illustrates this.

EXAMPLE 7 One grammar that generates the set $\{0^n1^n2^n \mid n = 0, 1, 2, 3, \ldots\}$ is $G = (V, T, S, P)$ with $V = \{0, 1, 2, S, A, B\}$, $T = \{0, 1, 2\}$, starting state S, and productions $S \rightarrow 0SAB$, $S \rightarrow \lambda$, $BA \rightarrow AB$, $0A \rightarrow 01$, $1A \rightarrow 11$, $1B \rightarrow 12$, $2B \rightarrow 22$. We leave it as an exercise for the reader to show that this statement is correct. The grammar given is the simplest type of grammar that generates this set, in a sense that will be made clear later in this section. The reader may wonder where this grammar came from, since it seems difficult to come up with this grammar from scratch. It may be comforting to know that this grammar can be systematically constructed using techniques from the theory of computation that are beyond the scope of this book. ■

TYPES OF PHRASE-STRUCTURE GRAMMARS

web

Phrase-structure grammars can be classified according to the types of productions that are allowed. We will describe the classification scheme introduced by Noam Chomsky. In Section 10.4 we will see that the different types of languages defined in this scheme correspond to the classes of languages that can be recognized using different models of computing machines.

A **type 0** grammar has no restrictions on its productions. A **type 1** grammar can have productions only of the form $w_1 \rightarrow w_2$, where the length of w_2 is greater than or equal to the length of w_1, or of the form $w_1 \rightarrow \lambda$. A **type 2** grammar can have productions only of the form $w_1 \rightarrow w_2$, where w_1 is a single symbol that is not a terminal symbol. A **type 3** grammar can have productions only of the form $w_1 \rightarrow w_2$ with $w_1 = A$ and either $w_2 = aB$ or $w_2 = a$, where A and B are nonterminal symbols and a is a terminal symbol, or with $w_1 = S$ and $w_2 = \lambda$.

From these definitions we see that every type 3 grammar is a type 2 grammar, every type 2 grammar is a type 1 grammar, and every type 1 grammar is a type 0 grammar. Type 2 grammars are called **context-free grammars** since a nonterminal symbol that is the left side of a production can be replaced in a string whenever it occurs, no matter what else is in the string. A language generated by a type 2 grammar is called a **context-free language.** When there is a production of the form $lw_1r \rightarrow lw_2r$ (but not of the form $w_1 \rightarrow w_2$), the grammar is called type 1 or **context-sensitive** since w_1 can be replaced by w_2 only when it is surrounded by the strings l and r. Type 3 grammars are also called **regular grammars.** A language generated by a regular grammar is called **regular.** Section 10.4 deals with the relationship between regular languages and finite-state machines. The Venn diagram in Figure 1 shows the relationship among different types of grammars.

EXAMPLE 8 From Example 6 we know that $\{0^m1^n \mid m, n = 0, 1, 2, \ldots\}$ is a regular language, since it can be generated by a regular grammar, namely, the grammar G_2 in Example 6. ■

---

web

Avram Noam Chomsky (born 1928). Noam Chomsky, born in Philadelphia, is the son of a Hebrew scholar. He received his B.A., M.A., and Ph.D. in linguistics, all from the University of Pennsylvania. He was on the staff of the University of Pennsylvania from 1950 until 1951. In 1955 he joined the faculty at M.I.T., beginning his M.I.T. career teaching engineers French and German. Chomsky is currently the Ferrari P. Ward Professor of foreign languages and linguistics at M.I.T. He is known for his many fundamental contributions to linguistics, including the study of grammars. Chomsky is also widely known for his outspoken political activism.

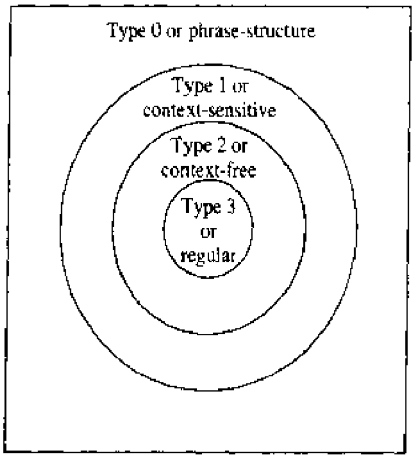


FIGURE 1 Types of Grammars.

EXAMPLE 9 It follows from Example 5 that $\{0^n1^n \mid n = 0, 1, 2, \ldots\}$ is a context-free language, since the productions in this grammar are $S \rightarrow 0S1$ and $S \rightarrow \lambda$. However, it is not a regular language. This will be shown in Section 10.4. ■

EXAMPLE 10 The set $\{0^n1^n2^n \mid n = 0, 1, 2, \ldots\}$ is a context-sensitive language, since it can be generated by a type 1 language, as Example 7 shows, but not by any type 2 language. (This is shown in Exercise 28 in the supplementary exercises at the end of the chapter.) ■

Table 1 summarizes the terminology used to classify phrase-structure grammars.

DERIVATION TREES

A derivation in the language generated by a context-free grammar can be represented graphically using an ordered rooted tree, called a **derivation,** or **parse, tree.** The root of this tree represents the starting symbol. The internal vertices of the tree represent the nonterminal symbols that arise in the derivation. The leaves of the tree represent the terminal symbols that arise. If the production $A \rightarrow w$ arises in the derivation, where w is a word, the vertex that represents A has as children vertices that represent each symbol in w, in order from left to right.

TABLE 1 Types of Grammars.

Type	*Restrictions on Productions* $w_1 \rightarrow w_2$
0	No restrictions
1	$l(w_1) \leq l(w_2)$, or $w_2 = \lambda$
2	$w_1 = A$ where A is nonterminal symbol
3	$w_1 = A$ and $w_2 = aB$ or $w_2 = a$, where $A \in N$, $B \in N$, and $a \in T$, or $S \rightarrow \lambda$

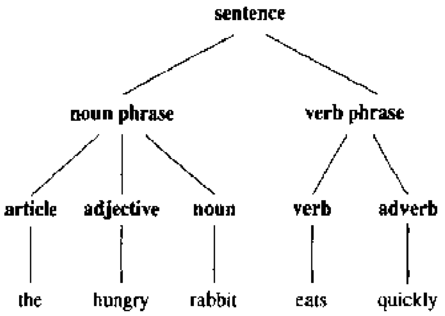


FIGURE 2 A Derivation Tree.

EXAMPLE 11 Construct a derivation tree for the derivation of *the hungry rabbit eats quickly*, given in the introduction of this section.

Solution: The derivation tree is shown in Figure 2. ■

The problem of determining whether a string is in the language generated by a context-free grammar arises in many applications, such as in the construction of compilers. Two approaches to this problem are indicated in the following example.

EXAMPLE 12 Determine whether the word *cbab* belongs to the language generated by the grammar $G = (V, T, S, P)$ where $V = \{a, b, c, A, B, C, S\}$, $T = \{a, b, c\}$, S is the starting symbol, and the productions are

$S \to AB$
$A \to Ca$
$B \to Ba$
$B \to Cb$
$B \to b$
$C \to cb$
$C \to b.$

Solution: One way to approach this problem is to begin with S and attempt to derive *cbab* using a series of productions. Since there is only one production with S on its left-hand side, we must start with $S \Rightarrow AB$. Next we use the only production that has A on its left-hand side, namely $A \to Ca$, to obtain $S \Rightarrow AB \Rightarrow CaB$. Since *cbab* begins with the symbols *cb*, we use the production $C \to cb$. This gives us $S \Rightarrow Ab \Rightarrow CaB \Rightarrow cbaB$. We finish by using the production $B \to b$, to obtain $S \Rightarrow AB \Rightarrow CaB \Rightarrow cbaB \Rightarrow cbab$. The approach that we have used is called **top-down parsing,** since it begins with the starting symbol and proceeds by successively applying productions.

There is another approach to this problem, called **bottom-up parsing.** In this approach, we work backward. Since *cbab* is the string to be derived, we can use the production $C \to cb$, so that $Cab \Rightarrow cbab$. Then, we can use the production $A \to Ca$, so that $Ab \Rightarrow Cab \Rightarrow cbab$. Using the production $B \to b$, gives $AB \Rightarrow Ab \Rightarrow Cab \Rightarrow cbab$. Finally, using $S \to AB$ shows that a complete derivation for *cbab* is $S \Rightarrow AB \Rightarrow Ab \Rightarrow Cab \Rightarrow cbab$. ■

BACKUS–NAUR FORM

There is another notation that is sometimes used to specify a type 2 grammar, called the **Backus–Naur form,** after John Backus, who invented it, and Peter Naur, who refined it for use in the specification of the programming language ALGOL. The Backus–Naur form is used to specify the syntactic rules of many computer languages, including Java. The productions in a type 2 grammar have a single nonterminal symbol as their left-hand side. Instead of listing all the productions separately, we can combine all those with the same nonterminal symbol on the left-hand side into one statement. Instead of using the symbol → in a production, we use the symbol $::=$. We enclose all nonterminal symbols in brackets, $\langle\ \rangle$, and we list all the right-hand sides of productions in the same statement, separating them by bars. For instance, the productions $A \to Aa$, $A \to a$, and $A \to AB$ can be combined into $\langle A \rangle ::= \langle A \rangle a \mid a \mid \langle A \rangle\langle B \rangle$.

EXAMPLE 13 What is the Backus–Naur form of the grammar for a subset of English described in the introduction to this section?

Solution: The Backus–Naur form of this grammar is:

$\langle \mathit{sentence} \rangle ::= \langle \mathit{noun\ phrase} \rangle\langle \mathit{verb\ phrase} \rangle$
$\langle \mathit{noun\ phrase} \rangle ::= \langle \mathit{article} \rangle\langle \mathit{adjective} \rangle\langle \mathit{noun} \rangle \mid \langle \mathit{article} \rangle\langle \mathit{noun} \rangle$
$\langle \mathit{verb\ phrase} \rangle ::= \langle \mathit{verb} \rangle\langle \mathit{adverb} \rangle \mid \langle \mathit{verb} \rangle$
$\langle \mathit{article} \rangle ::= a \mid \mathit{the}$
$\langle \mathit{adjective} \rangle ::= \mathit{large} \mid \mathit{hungry}$
$\langle \mathit{noun} \rangle ::= \mathit{rabbit} \mid \mathit{mathematician}$
$\langle \mathit{verb} \rangle ::= \mathit{eats} \mid \mathit{hops}$
$\langle \mathit{adverb} \rangle ::= \mathit{quickly} \mid \mathit{wildly}$ ■

EXAMPLE 14 Give the Backus–Naur form for the production of signed integers in decimal notation. (A **signed integer** is a nonnegative integer preceded by a plus sign or a minus sign.)

John Backus (born 1924). John Backus was born in Philadelphia. He received his bachelor of science and master's degree in mathematics from Columbia University. Backus joined IBM as a programmer in 1950. He participated in the design and development of two of IBM's early computers. From 1954 to 1958 he led the IBM group that developed FORTRAN. Backus became a staff member at the IBM Watson Research Center in 1958. He was part of the committees that designed the programming language ALGOL, using what is now called the Backus–Naur form for the description of the syntax of this language. Later, Backus worked on the mathematics of families of sets and on a functional style of programming. Backus became an IBM Fellow in 1963, and he received the National Medal of Science in 1974 and the prestigious Turing Award from the Association of Computing Machinery in 1977.

Peter Naur (born 1928). Peter Naur was born in Frederiksberg, near Copenhagen. As a boy he became interested in astronomy. Not only did he observe heavenly bodies, but he also computed the orbits of comets and asteroids. Naur attended Copenhagen University, receiving his degree in 1949. He spent 1950 and 1951 in Cambridge, where he used an early computer to calculate the motions of comets and planets. After returning to Denmark he continued working in astronomy but kept his ties to computing. In 1955 he served as a consultant to the building of the first Danish computer. In 1959 Naur made the switch from astronomy to computing as a full-time activity. His first job as a full-time computer scientist was participating in the development of the programming language ALGOL. From 1960 to 1967 he worked on the development of compilers for ALGOL and COBOL. In 1969 he became professor of computer science at Copenhagen University, where he has worked in the area of programming methodology.

Solution: The Backus–Naur form for a grammar that produces signed integers follows:

⟨*signed integer*⟩ ::= ⟨*sign*⟩⟨*integer*⟩
⟨*sign*⟩ ::= + | −
⟨*integer*⟩ ::= ⟨*digit*⟩ | ⟨*digit*⟩⟨*integer*⟩
⟨*digit*⟩ ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 ■

Exercises

Exercises 1–3 refer to the grammar with start symbol **sentence,** set of terminals T = {*the, sleepy, happy, tortoise, hare, passes, runs, quickly, slowly*}, set of nonterminals N = {**noun phrase, transitive verb phrase, intransitive verb phrase, article, adjective, noun, verb, adverb**}, and productions.

sentence → **noun phrase transitive verb phrase noun phrase**
sentence → **noun phrase intransitive verb phrase**
noun phrase → **article adjective noun**
noun phrase → **article noun**
transitive verb phrase → **transitive verb**
intransitive verb phrase → **intransitive verb adverb**
intransitive verb phrase → **intransitive verb**
article → *the*
adjective → *sleepy*
adjective → *happy*
noun → *tortoise*
noun → *hare*
transitive verb → *passes*
intransitive verb → *runs*
adverb → *quickly*
adverb → *slowly*

1. Use the set of productions to show that each of the following is a valid sentence.
   - a) *the happy hare runs*
   - b) *the sleepy tortoise runs quickly*
   - c) *the tortoise passes the hare*
   - d) *the sleepy hare passes the happy tortoise*
2. Find five other valid sentences, besides those given in Exercise 1.
3. Show that *the hare runs the sleepy tortoise* is not a valid sentence.

*4. Let $V = \{S, A, B, a, b\}$ and $T = \{a, b\}$. Find the language generated by the grammar (V, T, S, P) when the set P of products consists of
   - a) $S \rightarrow AB$, $A \rightarrow ab$, $B \rightarrow bb$.
   - b) $S \rightarrow AB$, $S \rightarrow aA$, $A \rightarrow a$, $B \rightarrow ba$.
   - c) $S \rightarrow AB$, $S \rightarrow AA$, $A \rightarrow aB$, $A \rightarrow ab$, $B \rightarrow b$.
   - d) $S \rightarrow AA$, $S \rightarrow B$, $A \rightarrow aaA$, $A \rightarrow aa$, $B \rightarrow bB$, $B \rightarrow b$.
   - e) $S \rightarrow AB$, $A \rightarrow aAb$, $B \rightarrow bBa$, $A \rightarrow \lambda$, $B \rightarrow \lambda$.

5. Construct a derivation of $0^3 1^3$ using the grammar given in Example 5.
6. Show that the grammar given in Example 5 generates the set $\{0^n 1^n \mid n = 0, 1, 2, \ldots\}$.
7.
   - a) Construct a derivation of $0^2 1^4$ using the grammar G_1 in Example 6.
   - b) Construct a derivation of $0^2 1^4$ using the grammar G_2 in Example 6.
8.
   - a) Show that the grammar G_1 given in Example 6 generates the set $\{0^m 1^n \mid m, n = 0, 1, 2, \ldots\}$.
   - b) Show that the grammar G_2 in Example 6 generates the same set.
9. Construct a derivation of $0^2 1^2 2^2$ in the grammar given in Example 7.

*10. Show that the grammar given in Example 7 generates the set $\{0^n 1^n 2^n \mid n = 0, 1, 2, \ldots\}$.

*11. Find a phrase-structure grammar for each of the following languages.
   - a) the set of all bit strings containing an even number of 0s and no 1s
   - b) the set of all bit strings made up of a 1 followed by an odd number of 0s
   - c) the set of all bit strings containing an even number of 0s and an even number of 1s
   - d) the set of all strings containing 10 or more 0s and no 1s
   - e) the set of all strings containing more 0s than 1s
   - f) the set of all strings containing an equal number of 0s and 1s
   - g) the set of all strings containing an unequal number of 0s and 1s

12. Construct phrase-structure grammars to generate each of the following sets.
    - a) $\{01^{2n} \mid n \geq 0\}$
    - b) $\{0^n 1^{2n} \mid n \geq 0\}$
    - c) $\{0^n 1^m 0^n \mid m \geq 0, n \geq 0\}$
13. Let $V = \{S, A, B, a, b\}$ and $T = \{a, b\}$. Determine whether $G = (V, T, S, P)$ is a type 0 grammar but not a type 1 grammar, a type 1 grammar but not a type 2 grammar, or a type 2 grammar but not a type 3 grammar if P, the set of productions, is

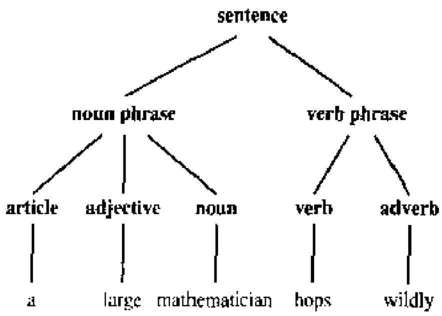
a) $S \to aAB, A \to Bb, B \to \lambda$.
b) $S \to aA, A \to a, A \to b$.
c) $S \to ABa, AB \to a$.
d) $S \to ABA, A \to aB, B \to ab$.
e) $S \to bA, A \to B, B \to a$.
f) $S \to aA, aA \to B, B \to aA, A \to b$.
g) $S \to bA, A \to b, S \to \lambda$.
h) $S \to AB, B \to aAb, aAb \to b$.
i) $S \to aA, A \to bB, B \to b, B \to \lambda$.
j) $S \to A, A \to B, B \to \lambda$.

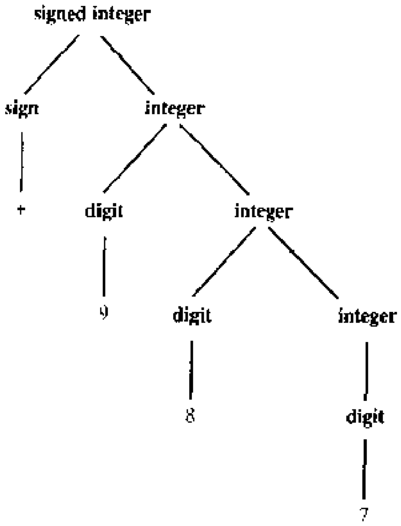
14. A **palindrome** is a string that reads the same backward as it does forward, that is, a string w where $w = w^R$, where w^R is the reversal of the string w. Find a context-free grammar that generates the set of all palindromes over the alphabet $\{0, 1\}$.

***15.** Let G_1 and G_2 be context-free grammars, generating the languages $L(G_1)$ and $L(G_2)$, respectively. Show that there is a context-free grammar generating each of the following sets.

a) $L(G_1) \cup L(G_2)$ **b)** $L(G_1)L(G_2)$
c) $L(G_1)^*$

16. Find the strings constructed using the derivation trees shown below.





17. Construct derivation trees for the sentences in Exercise 1.

18. Let G be the grammar with $V = \{a, b, c, S\}$, $T = \{a, b, c\}$, starting symbol S, and productions $S \to abS$, $S \to bcS$, $S \to bbS$, $S \to a$, $S \to cb$. Construct derivation trees for

a) $bcbba$.
b) $bbbcbba$.
c) $bcabbbbbcb$.

***19.** Use top-down parsing to determine whether each of the following strings belongs to the language generated by the grammar in Example 12.

a) $baba$ **b)** $abab$
c) $cbaba$ **d)** $bbbcba$

***20.** Use bottom-up parsing to determine whether the strings in Exercise 19 belong to the language generated by the grammar in Example 12.

21. Construct a derivation tree for -109 using the grammar given in Example 14.

22. **a)** Explain what the productions are in a grammar if the Backus–Naur form for productions is as follows?

$$\begin{aligned} \langle expression \rangle ::= {} & (\langle expression \rangle) \mid \\ & \langle expression \rangle + \langle expression \rangle \mid \\ & \langle expression \rangle * \langle expression \rangle \\ & \langle variable \rangle \\ \langle variable \rangle ::= {} & x \mid y \end{aligned}$$

b) Find a derivation tree for $(x * y) + x$ in this grammar.

23. **a)** Construct a phrase-structure grammar that generates all signed decimal numbers, consisting of a sign, either $+$ or $-$; a nonnegative integer; and a decimal fraction that is either the empty string or a decimal point followed by a positive integer, where initial zeros in an integer are allowed.

b) Give the Backus–Naur form of this grammar.

c) Construct a derivation tree for -31.4 in this grammar.

24. **a)** Construct a phrase-structure grammar for the set of all fractions of the form a/b, where a is a signed integer in decimal notation and b is a positive integer.

b) What is the Backus–Naur form for this grammar?

c) Construct a derivation tree for $+311/17$ in this grammar.

25. Let G be a grammar and let R be the relation containing the ordered pair (w_0, w_1) if and only if w_1 is directly derivable from w_0 in G. What is the reflexive transitive closure of R?

10.2

Finite-State Machines with Output

INTRODUCTION

Many kinds of machines, including components in computers, can be modeled using a structure called a finite-state machine. Several types of finite-state machines are commonly used in models. All these versions of finite-state machines include a finite set of states, with a designated starting state, an input alphabet, and a transition function that assigns a next state to every state and input pair. In this section we will study those finite-state machines that produce output. We will show how finite-state machines can be used to model a vending machine, a machine that delays input, a machine that adds integers, and a machine that determines whether a bit string contains a specified pattern.

Before giving formal definitions, we will show how a vending machine can be modeled. A vending machine accepts nickels (5 cents), dimes (10 cents), and quarters (25 cents). When a total of 30 cents or more has been deposited, the machine immediately returns the amount in excess of 30 cents. When 30 cents has been deposited and any excess refunded, the customer can push an orange button and receive an orange juice or push a red button and receive an apple juice. We can describe how the machine works by specifying its states, how it changes states when input is received, and the output that is produced for every combination of input and current state.

The machine can be in any of seven different states s_i, $i = 0, 1, 2, \ldots, 6$, where s_i is the state where the machine has collected $5i$ cents. The machine starts in state s_0, with 0 cents received. The possible inputs are 5 cents, 10 cents, 25 cents, the orange button (O), and the red button (R). The possible outputs are nothing (n), 5 cents, 10 cents, 15 cents, 20 cents, 25 cents, an orange juice, and an apple juice.

We illustrate how this model of the machine works with the following example. Suppose that a student puts in a dime followed by a quarter, receives 5 cents back, and then pushes the orange button for an orange juice. The machine starts in state s_0. The first input is 10 cents, which changes the state of the machine to s_2 and gives no output. The second input is 25 cents. This changes the state from s_2 to s_6, and gives 5 cents as output. The next input is the orange button, which changes the state from s_6 back to s_0 (since the machine returns to the start state) and gives an orange juice as its output.

We can display all the state changes and output of this machine in a table. To do this we need to specify for each combination of state and input the next state and the output obtained. Table 1 shows the transitions and outputs for each pair of a state and an input.

Another way to show the actions of a machine is to use a directed graph with labeled edges, where each state is represented by a circle, edges represent the transitions, and edges are labeled with the input and the output for that transition. Figure 1 shows such a directed graph for the vending machine.

FINITE-STATE MACHINES WITH OUTPUTS

We will now give the formal definition of a finite-state machine with output.

TABLE 1 **State Table for a Vending Machine.**

	Next State					*Output*				
	Input					*Input*				
State	*5*	*10*	*25*	*O*	*R*	*5*	*10*	*25*	*O*	*R*
s_0	s_1	s_2	s_5	s_0	s_0	n	n	n	n	n
s_1	s_2	s_3	s_6	s_1	s_1	n	n	n	n	n
s_2	s_3	s_4	s_6	s_2	s_2	n	n	5	n	n
s_3	s_4	s_5	s_6	s_3	s_3	n	n	10	n	n
s_4	s_5	s_6	s_6	s_4	s_4	n	n	15	n	n
s_5	s_6	s_6	s_6	s_5	s_5	n	5	20	n	n
s_6	s_6	s_6	s_6	s_0	s_0	5	10	25	OJ	AJ

DEFINITION 1. A *finite-state machine* $M = (S, I, O, f, g, s_0)$ **consists of a finite set** S of *states*, a finite *input alphabet* I, a finite *output alphabet* O, a *transition function* f that assigns to each state and input pair a new state, an *output function* g that assigns to each state and input pair an output, and an **initial state** s_0.

Let $M = (S, I, O, f, g, s_0)$ be a finite-state machine. We can use a **state table** to represent the values of the transition function f and the output function g for all pairs of states and input. We previously constructed a state table for the vending machine discussed in the introduction to this section.

EXAMPLE 1 The state table shown in Table 2 describes a finite-state machine with $S = \{s_0, s_1, s_2, s_3\}$, $I = \{0, 1\}$, and $O = \{0, 1\}$. The values of the transition function f are displayed in the first two columns, and the values of the output function g are displayed in the last two columns. ■

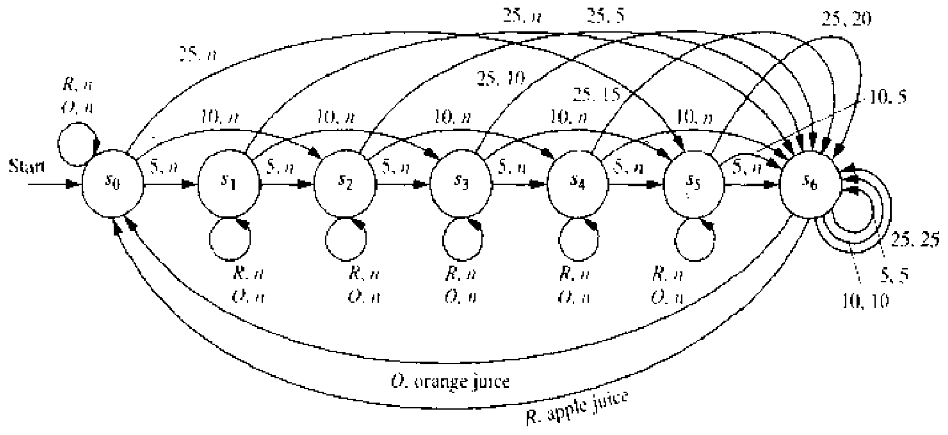


FIGURE 1 **A Vending Machine.**

TABLE 2

	f		*g*	
	Input		*Input*	
State	*0*	*1*	*0*	*1*
s_0	s_1	s_0	1	0
s_1	s_3	s_0	1	1
s_2	s_1	s_2	0	1
s_3	s_2	s_1	0	0

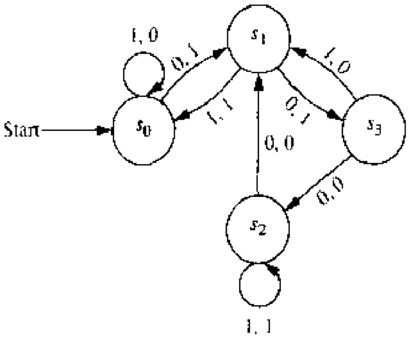


FIGURE 2 The State Diagram for the Finite-State Machine Shown in Table 2.

Another way to represent a finite-state machine is to use a **state diagram,** which is a directed graph with labeled edges. In this diagram, each state is represented by a circle. Arrows labeled with the input and output pair are shown for each transition.

EXAMPLE 2 Construct the state diagram for the finite-state machine with the state table shown in Table 2.

Solution: The state diagram for this machine is shown in Figure 2. ■

EXAMPLE 3 Construct the state table for the finite-state machine with the state diagram shown in Figure 3.

Solution: The state table for this machine is shown in Table 3. ■

An input string takes the starting state through a sequence of states, as determined by the transition function. As we read the input string symbol by symbol (from left to right), each input symbol takes the machine from one state to another. Because each transition produces an output, an input string also produces an output string.

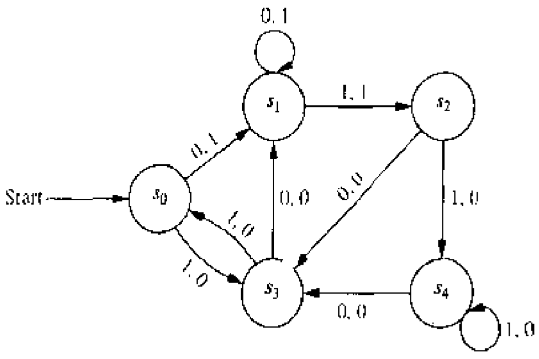


FIGURE 3 A Finite-State Machine.

TABLE 3

	f		*g*	
	Input		*Input*	
State	*0*	*1*	*0*	*1*
s_0	s_1	s_3	1	0
s_1	s_1	s_2	1	1
s_2	s_3	s_4	0	0
s_3	s_1	s_0	0	0
s_4	s_3	s_4	0	0

Suppose that the input string is $x = x_1x_2 \cdots x_k$. Then, reading this input takes the machine from state s_0 to state s_1, where $s_1 = f(s_0, x_1)$, then to state s_2, where $s_2 = f(s_1, x_2)$, and so on, ending at state $s_k = f(s_{k-1}, x_k)$. This sequence of transitions produces an output string $y_1y_2 \cdots y_k$, where $y_1 = g(s_0, x_1)$ is the output corresponding to the transition from s_0 to s_1, $y_2 = g(s_1, x_2)$ is the output corresponding to the transition from s_1 to s_2, and so on. In general $y_j = g(s_{j-1}, x_j)$ for $j = 1, 2, \ldots, k$. Hence, we can extend the definition of the output function g to input strings so that $g(x) = y$, where y is the output corresponding to the input string x. This notation is useful in many applications.

EXAMPLE 4 Find the output string generated by the finite-state machine in Figure 3 if the input string is 101011.

Solution: The output obtained is 001000. The successive states and outputs are shown in Table 4. ■

We can now give some examples of useful finite-state machines. These examples illustrate that the states of a finite-state machine give it limited memory capabilities. The states can be used to remember the properties of the symbols that have been read by the machine. However, since there are only finitely many different states, finite-state machines cannot be used for some important purposes. This will be illustrated in Section 10.4.

EXAMPLE 5 An important element in many electronic devices is a *unit-delay machine*, which produces as output the input string delayed by a specified amount of time. How can a finite-state machine be constructed that delays an input string by one unit of time, that is, produces as output the bit string $0x_1x_2 \cdots x_{k-1}$ given the input bit string $x_1x_2 \cdots x_k$?

Solution: A delay machine can be constructed that has two possible inputs, namely, 0 and 1. The machine must have a start state s_0. Since the machine has to remember whether the previous input was a 0 or a 1, two other states s_1 and s_2 are needed, where the machine is in state s_1 if the previous input was 1 and in state s_2 if the previous input was 0. An output of 0 is produced for the initial transition from s_0. Each transition from s_1 gives an output of 1, and each transition from s_2 gives an output of 0. The output corresponding to the input of a string $x_1 \cdots x_k$ is the string that begins with 0, followed by x_1, followed by $x_2, \ldots,$ ending with x_{k-1}. The state diagram for this machine is shown in Figure 4. ■

TABLE 4

Input	1	0	1	0	1	1	–
State	s_0	s_3	s_1	s_2	s_3	s_0	s_3
Output	0	0	1	0	0	0	–

EXAMPLE 6 Produce a finite-state machine that adds two integers using their binary expansions.

Solution: When $(x_n \cdots x_1x_0)_2$ and $(y_n \cdots y_1y_0)_2$ are added, the following procedure (as described in Section 2.4) is followed. First, the bits x_0 and y_0 are added, producing a sum bit z_0 and a carry bit c_0. This carry bit is either 0 or 1. Then, the bits x_1 and y_1 are added, together with the carry c_0. This gives a sum bit z_1 and a carry bit c_1. This procedure is continued until the nth stage, where x_n, y_n, and the previous carry c_{n-1} are added to produce the sum bit z_n and the carry bit c_n, which is equal to the sum bit z_{n-1}.

A finite-state machine to carry out this addition can be constructed using just two states. For simplicity we assume that both the initial bits x_n and y_n are 0 (otherwise we have to make special arrangements concerning the sum bit z_{n+1}). The start state s_0 is used to remember that the previous carry is 0 (or for the addition of the rightmost bits). The other state, s_1, is used to remember that the previous carry is 1. Since the inputs to the machine are pairs of bits, there are four possible inputs. We represent these possibilities by 00 (when both bits are 0), 01 (when the first bit is 0 and the second is 1), 10 (when the first bit is 1 and the second is 0), and 11 (when both bits are 1). The transitions and the outputs are constructed from the sum of the two bits represented by the input and the carry represented by the state. For instance, when the machine is in state s_1 and receives 01 as input, the next state is s_1 and the output is 0, since the sum that arises is $0+1+1 = (10)_2$. The state diagram for this machine is shown in Figure 5. ■

EXAMPLE 7 In a certain coding scheme, when three consecutive 1s appear in a message, the receiver of the message knows that there has been a transmission error. Construct a finite-state machine that gives a 1 as its output bit if and only if the last three bits received are all 1s.

Solution: Three states are needed in this machine. The start state s_0 remembers that the previous input value, if it exists, was not a 1. The state s_1 remembers that the previous input was a 1, but the input before the previous input, if it exists, was not a 1. The state s_2 remembers that the previous two inputs were 1s. An input of 1 takes s_0 to s_1, since now a 1, and not two consecutive 1s, has been read; it takes s_1 to s_2, since now two consecutive 1s have been read; and it takes s_2 to itself, since at least two consecutive 1s have been read. An input of 0 takes every state to s_0, since this breaks up any string of consecutive 1s. The output for the transition from s_2 to itself when a 1 is read is 1, since this combination of input and state show that three consecutive 1s have been read. All other outputs are 0. The state diagram of this machine is shown in Figure 6. ■

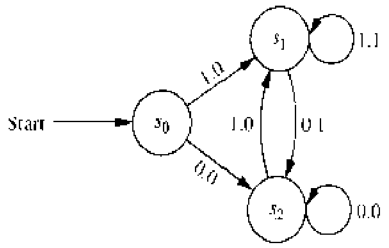


FIGURE 4 A Unit-Delay Machine.

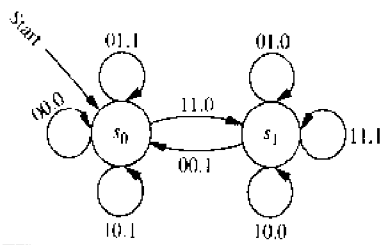


FIGURE 5 A Finite-State Machine for Addition.

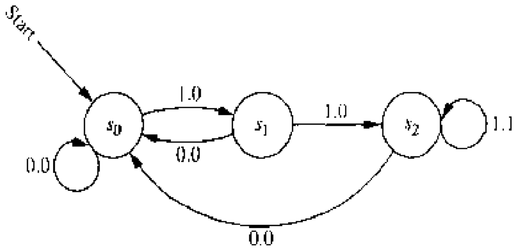


FIGURE 6 A Finite-State Machine That Gives an Output of 1 If and Only If the Input String Read So Far Ends with 111.

The machine in Figure 6 is an example of a **language recognizer,** because it produces an output of 1 if and only if the input string read so far has a specified property. Language recognition is an important application of finite-state machines.

Types of Finite-State Machines Many different kinds of finite-state machines have been developed to model computing machines. In this section we have given a definition of one type of finite-state machine. In the type of machine introduced in this section, outputs correspond to transitions between states. Machines of this type are known as **Mealy machines,** since they were first studied by G. H. Mealy in 1955. There is another important type of finite-state machine with output, where the output is determined only by the state. This type of finite-state machine is known as a **Moore machine,** since E. F. Moore introduced this type of machine in 1956. Moore machines are considered in a sequence of exercises at the end of this section.

In Example 7 we showed how a Mealy machine can be used for language recognition. However, another type of finite-state machine, giving no output, is usually used for this purpose. Finite-state machines with no output, also known as finite-state automata, have a set of final states and recognize a string if and only if it takes the start state to a final state. We will study this type of finite-state machine in Section 10.3.

Exercises

1. Draw the state diagrams for the finite-state machines with the following state tables.

a)

	f		g	
State	***Input*** ***0***	***1***	***Input*** ***0***	***1***
s_0	s_1	s_0	0	1
s_1	s_0	s_2	0	1
s_2	s_1	s_1	0	0

b)

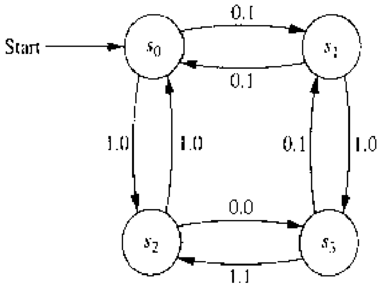
	f		g	
State	***Input*** ***0***	***1***	***Input*** ***0***	***1***
s_0	s_1	s_0	0	0
s_1	s_2	s_0	1	1
s_2	s_0	s_3	0	1
s_3	s_1	s_2	1	0

c)

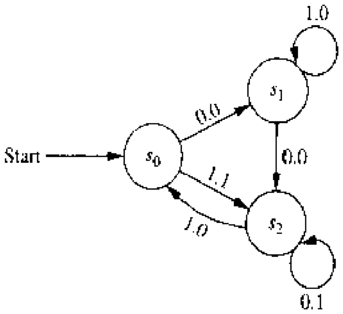
	f		g	
State	***Input*** ***0***	***1***	***Input*** ***0***	***1***
s_0	s_0	s_4	1	1
s_1	s_0	s_3	0	1
s_2	s_0	s_2	0	0
s_3	s_1	s_1	1	1
s_4	s_1	s_0	1	0

2. Give the state tables for the finite-state machines with the following state diagrams.

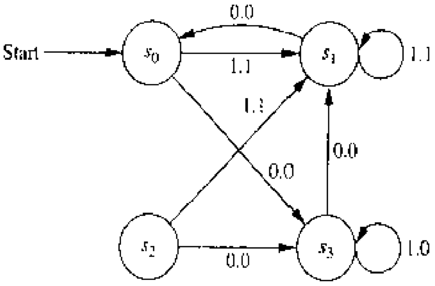
a)



b)



c)



3. Given the finite-state machine shown in Example 2, determine the output for each of the following input strings.
   a) 0111 b) 11011011 c) 01010101010
4. Given the finite-state machine shown in Example 3, determine the output for each of the following input strings.
   a) 0000 b) 101010 c) 11011100010
5. Construct a finite-state machine that models a soda machine which accepts nickels, dimes, and quarters. The soda machine accepts change until 35 cents has been put in. It gives change back for any amount greater than 35 cents. Then the customer can push buttons to receive either a cola, a root beer, or a ginger ale.
6. Construct a finite-state machine that models a newspaper vending machine which has a door that can be opened only after either three dimes (and any number of other coins) or a quarter and a nickel (and any number of other coins) have been inserted. Once the door can be opened, the customer opens it and takes a paper, closing the door. No change is ever returned no matter how much extra money has been inserted. The next customer starts with no credit.
7. Construct a finite-state machine that delays an input string two bits, giving 00 as the first two bits of output.
8. Construct a finite-state machine that changes every other bit, starting with the second bit, of an input string, and leaves the other bits unchanged.
9. Construct a finite-state machine for the log-on procedure for a computer, where the user logs in by entering a user identification number, which is considered to be a single input, and then a password, which is considered to be a single input. If the password is incorrect, the user is asked for the user identification number again.
10. Construct a finite-state machine for a combination lock that contains numbers 1 through 40 and that opens only when the correct combination, 10 right, 8 second left, 37 right, is entered. Each input is a triple consisting of a number, the direction of the turn, and the number of times the lock is turned in that direction.
11. Construct a finite-state machine for a toll machine that opens a gate after 25 cents, in nickels, dimes, or quarters, has been deposited. No change is given for overpayment, and no credit is given to the next driver when more than 25 cents has been deposited.
12. Construct a finite-state machine that gives an output of 1 if the number of input symbols read so far is divisible by 3 and an output of 0 otherwise.
13. Construct a finite-state machine that determines whether the input string has a 1 in the last position and a 0 in the third to the last position read so far.
14. Construct a finite-state machine that determines whether the input string read so far ends in at least five consecutive 1s.
15. Construct a finite-state machine that determines whether the word *computer* has been read as the last eight characters in the input read so far, where the input can be any string of English letters.

A **Moore machine** $M = (S, I, O, f, g, s_0)$ consists of a finite set of states, an input alphabet I, an output alphabet O, a transition function f that assigns a next state to every pair of a state and an input, an output function g that assigns an output to every state, and a starting state s_0. A Moore machine can be represented either by a table listing the transitions for each pair of state and input and the outputs for each state, or by a state diagram that displays the states, the transitions between states, and the output for each state. In the diagram, transitions are indicated with arrows labeled with the input, and the outputs are shown next to the states.

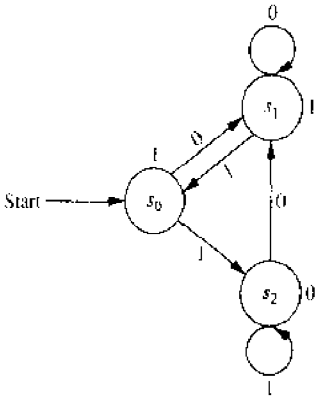
16. Construct the state diagram for the Moore machine with the following state table.

State	*f* Input 0	1	*g*
s_0	s_0	s_2	0
s_1	s_3	s_0	1
s_2	s_2	s_1	1
s_3	s_2	s_0	1

17. Construct the state table for the Moore machine with the state diagram shown at the top of the next column. Each input string to a Moore machine M produces an output string. In particular, the output corresponding to the input string $a_1a_2\cdots a_k$ is the string $g(s_0)g(s_1)\ldots g(s_k)$ where $s_i = f(s_{i-1}, a_i)$ for $i = 1, 2, \ldots, k$.

18. Find the output string generated by the Moore machine in Exercise 16 with each of the following input strings.
   a) 0101 b) 111111 c) 11101110111

19. Find the output string generated by the Moore machine in Exercise 17 with each of the input strings in Exercise 18.



20. Construct a Moore machine that gives an output of 1 whenever the number of symbols in the input string read so far is divisible by 4.

21. Construct a Moore machine that determines whether an input string contains an even or odd number of 1s. The machine should give 1 as output if an even number of 1s are in the string and 0 as output if an odd number of 1s are in the string.

10.3

Finite-State Machines with No Output

INTRODUCTION

web

One of the most important applications of finite-state machines is in language recognition. This application plays a fundamental role in the design and construction of compilers for programming languages. In Section 10.2 we showed that a finite-state machine with output can be used to recognize a language, by giving an output of 1 when a string from the language has been read and a 0 otherwise. However, there are other types of finite-state machines that are specially designed for recognizing languages. Instead of producing output, these machines have final states. A string is recognized if and only if it takes the starting state to one of these final states.

SET OF STRINGS

Before discussing finite-state machines with no output, we will introduce some important background material on sets of strings. The operations that will be defined here will be used extensively in our discussion of language recognition by finite-state machines.

DEFINITION 1. Suppose that A and B are subsets of V^*, where V is a vocabulary. The **concatenation** of A and B, denoted by AB, is the set of all strings of the form xy where x is a string in A and y is a string in B.

EXAMPLE 1 Let $A = \{0, 11\}$ and $B = \{1, 10, 110\}$. Find AB and BA.

Solution: The set AB contains every concatenation of a string in A and a string in B. Hence, $AB = \{01, 010, 0110, 111, 1110, 11110\}$. The set BA contains every concatenation of a string in B and a string in A. Hence, $BA = \{10, 111, 100, 1011, 1100, 11011\}$. ■

Note that it is not necessarily the case that $AB = BA$ when A and B are subsets of V^*, where V is an alphabet, as Example 1 illustrates.

From the definition of the concatenation of two sets of strings, we can define A^n, for $n = 0, 1, 2, \ldots$. This is done recursively by specifying that

$$A^0 = \{\lambda\},$$

$$A^{n+1} = A^nA \qquad \text{for } n = 0, 1, 2, \ldots.$$

EXAMPLE 2 Let $A = \{1, 00\}$. Find A^n for $n = 0, 1, 2,$ and 3.

Solution: We have $A^0 = \{\lambda\}$ and $A^1 = A^0A = \{\lambda\}A = \{1, 00\}$. To find A^2 we take concatenations of pairs of elements of A. This gives $A^2 = \{11, 100, 001, 0000\}$. To find A^3 we take concatenations of elements in A^2 and A; this gives $A^3 = \{111, 1100, 1001, 10000, 0011, 00100, 00001, 000000\}$. ■

DEFINITION 2. Suppose that A is a subset of V^*. Then the *Kleene closure* of A, denoted by A^*, is the set consisting of concatenations of arbitrarily many strings from A. That is, $A^* = \bigcup_{k=0}^{\infty} A^k$.

EXAMPLE 3 What are the Kleene closures of the sets $A = \{0\}$, $B = \{0, 1\}$, and $C = \{11\}$?

Solution: The Kleene closure of A is the concatenation of the string 0 with itself an arbitrary finite number of times. Hence $A^* = \{0^n \mid n = 0, 1, 2, \ldots\}$. The Kleene closure of B is the concatenation of an arbitrary number of strings where each string is either 0 or

---

Stephen Cole Kleene (1909–1994). Stephen Kleene was born in Hartford, Connecticut. His mother, Alice Lena Cole, was a poet, and his father, Gustav Adolph Kleene, was an economics professor. Kleene attended Amherst College and received his Ph.D. from Princeton in 1934, where he studied under the famous logician Alonzo Church. Kleene joined the faculty of the University of Wisconsin in 1935, where he remained except for several leaves, including stays at the Institute for Advanced Study in Princeton. During World War II he was a navigation instructor at the Naval Reserve's Midshipmen's School and later served as the director of the Naval Research Laboratory. Kleene made significant contributions to the theory of recursive functions, investigating questions of computability and decidability, and proved one of the central results of automata theory. He served as the Acting Director of the Mathematics Research Center and as Dean of the College of Letters and Sciences at the University of Wisconsin. Kleene was a student of natural history. He discovered a previously undescribed variety of butterfly that is named after him. He was an avid hiker and climber. Kleene was also noted as a talented teller of anecdotes, using a powerful voice that could be heard several offices away.

1. This is the set of all strings over the alphabet $V = \{0, 1\}$. That is, $B^* = V^*$. Finally, the Kleene closure of C is the concatenation of the string 11 with itself an arbitrary number of times. Hence, C^* is the set of strings consisting of an even number of 1s. That is, $C^* = \{1^{2n} \mid n = 0, 1, 2, \ldots\}$. ■

FINITE-STATE AUTOMATA

We will now give a definition of a finite-state machine with no output. Such machines are also called **finite-state automata,** and that is the terminology we will use for them here. (*Note:* The singular of *automata* is *automaton.*) These machines differ from the finite-state machines studied in Section 10.2 in that they do not produce output, but they do have a set of final states. As we will see, they recognize strings that take the starting state to a final state.

DEFINITION 3. **A *finite-state automaton* $M = (S, I, f, s_0, F)$ consists of a finite set S of *states,* a finite *input alphabet* I, a *transition function* f that assigns a next state to every pair of state and input, an initial state s_0, and a subset F of S consisting of *final states.***

We can represent finite-state automata using either state tables or state diagrams. Final states are indicated in state diagrams by using double circles.

EXAMPLE 4 Construct the state diagram for the finite-state automaton $M = (S, I, f, s_0, F)$, where $S = \{s_0, s_1, s_2, s_3\}$, $I = \{0, 1\}$, $F = \{s_0, s_3\}$, and the transition function f is given in Table 1.

Solution: The state diagram is shown in Figure 1. Note that since both the inputs 0 and 1 take s_2 to s_0, we write 0,1 over the edge from s_2 to s_0. ■

The transition function f can be extended so that it is defined for all pairs of states and strings. Let $x = x_1x_2 \cdots x_k$ be a string in I^*. Then $f(s_1, x)$ is the state obtained by using each successive symbol of x, from left to right, as input, starting with state s_1.

TABLE 1

	f	
	Input	
State	***0***	***1***
s_0	s_0	s_1
s_1	s_0	s_2
s_2	s_0	s_0
s_3	s_2	s_1

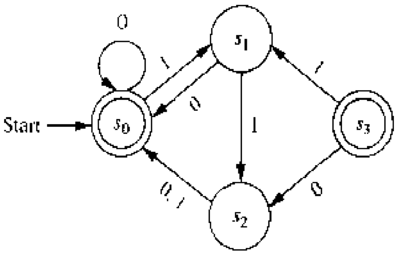


FIGURE 1 **The State Diagram for a Finite-State Automaton.**

From s_1 we go on to state $s_2 = f(s_1, x_1)$, then to state $s_2 = f(s_2, x_2)$, and so on, with $f(s_1, x) = f(s_k, x_k)$.

A string x is said to be **recognized** or **accepted** by the machine $M = (S, I, f, s_0, F)$ if it takes the initial state s_0 to a final state, that is, $f(s_0, x)$ is a state in F. The **language recognized** or **accepted** by the machine M, denoted by $L(M)$, is the set of all strings that are recognized by M. Two finite-state automata are called **equivalent** if they recognize the same language.

EXAMPLE 5 Determine the language recognized by the finite-state automata M_1, M_2, and M_3 in Figure 2.

Solution: The only final state of M_1 is s_0. The strings that take s_0 to itself are those consisting of zero or more consecutive 1s. Hence, $L(M_1) = \{1^n \mid n = 0, 1, 2, \ldots\}$.

The only final state of M_2 is s_2. The only strings that take s_0 to s_2 are 1 and 01. Hence, $L(M_2) = \{1, 01\}$.

The final states of M_3 are s_0 and s_3. The only strings that take s_0 to itself are $\lambda, 0, 00, 000, \ldots$, that is, any string of zero or more consecutive 0s. The only strings that take s_0 to s_3 are a string of zero or more consecutive 0s, followed by 10, followed by any string. Hence, $L(M_3) = \{0^n, 0^n10x \mid n = 0, 1, 2, \ldots, \text{and } x \text{ is any string}\}$. ■

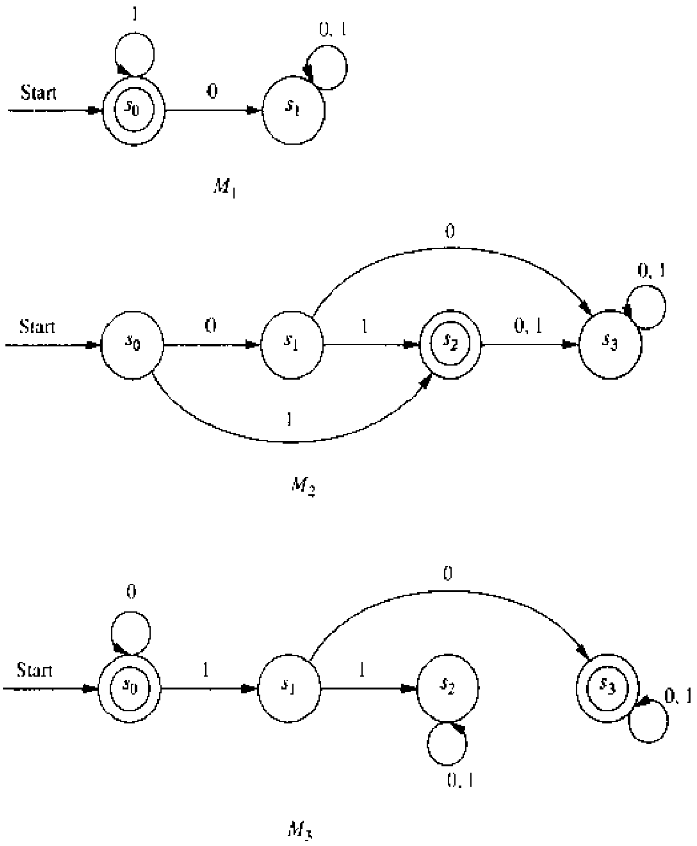


FIGURE 2 Some Finite-State Automata.

The finite-state automata discussed so far are **deterministic,** since for each pair of state and input value there is a unique next state given by the transition function. There is another important type of finite-state automaton in which there may be several possible next states for each pair of input value and state. Such machines are called **nondeterministic.** Nondeterministic finite-state automata are important in determining which languages can be recognized by a finite-state automaton.

DEFINITION 4. A ***nondeterministic finite-state automaton*** $M = (S, I, f, s_0, F)$ **consists of a set S of states, an input alphabet I, a transition function f that assigns a set of states to each pair of state and input, a starting state s_0, and a subset F of S consisting of the final states.**

We can represent nondeterministic finite-state automata using state tables or state diagrams. When we use a state table, for each pair of state and input values we give a list of possible next states. In the state diagram we include an edge from each state to all possible next states, labeling edges with the input or inputs that lead to this transition.

EXAMPLE 6 Find the state diagram for the nondeterministic finite-state automaton with the state table shown in Table 2. The final states are s_2 and s_3.

Solution: The state diagram for this automaton is shown in Figure 3. ■

EXAMPLE 7 Find the state table for the nondeterministic finite-state automaton with the state diagram shown in Figure 4.

Solution: The state table is given as Table 3. ■

What does it mean for a nondeterministic finite-state automaton to recognize a string $x = x_1x_2 \cdots x_k$? The first input symbol x_1 takes the starting state s_0 to a set S_1 of states. The next input symbol x_2 takes each of the states in S_1 to a set of states. Let

TABLE 2

	f	
	Input	
State	*0*	*1*
s_0	s_0, s_1	s_3
s_1	s_0	s_1, s_3
s_2		s_0, s_2
s_3	s_0, s_1, s_2	s_1

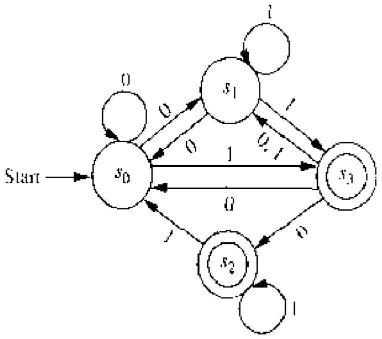


FIGURE 3 The Nondeterministic Finite-State Automaton with State Table Given in Table 2.

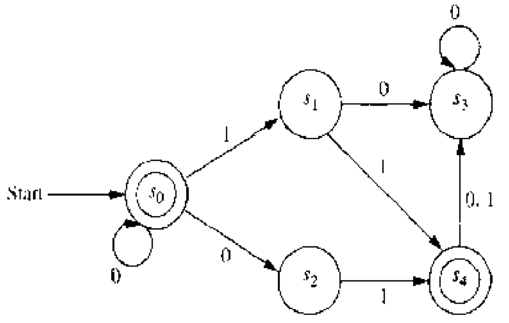


FIGURE 4 A Nondeterministic Finite-State Automaton.

TABLE 3

State	f Input 0	f Input 1
s_0	s_0, s_2	s_1
s_1	s_3	s_4
s_2		s_4
s_3	s_3	
s_4	s_3	s_3

S_2 be the union of these sets. We continue this process, including at a stage all states obtained using a state obtained at the previous stage and the current input symbol. We **recognize,** or **accept,** the string x if there is a final state in the set of all states that can be obtained from s_0 using x. The **language recognized** by a nondeterministic finite-state automaton is the set of all strings recognized by this automaton.

EXAMPLE 8 Find the language recognized by the nondeterministic finite-state automaton shown in Figure 4.

Solution: Since s_0 is a final state, and there is a transition from s_0 to itself when 0 is the input, the machine recognizes all strings consisting of zero or more consecutive 0s. Furthermore, since s_4 is a final state, any string that has s_4 in the set of states that can be reached from s_0 with this input string is recognized. The only such strings are strings consisting of zero or more consecutive 0s followed by 01 or 11. Since s_0 and s_4 are the only final states, the language recognized by the machine is $\{0^n, 0^n01, 0^n11 \mid n \geq 0\}$. ∎

One important fact is that a language recognized by a nondeterministic finite-state automaton is also recognized by a deterministic finite-state automaton. We will take advantage of this fact in the next section when we will determine which languages are recognized by finite-state automata.

THEOREM 1 **If the language L is recognized by a nondeterministic finite-state automaton M_0, then L is also recognized by a deterministic finite-state automaton M_1.**

Proof: We will describe how to construct the deterministic finite-state automaton M_1 that recognizes L from M_0, the nondeterministic finite-state automaton that recognizes this language. Each state in M_1 will be made up of a set of states in M_0. The start symbol of M_1 is $\{s_0\}$, which is the set containing the start state of M_0. The input set of M_1 is the same as the input set of M_0. Given a state $\{s_{i_1}, s_{i_2}, \ldots, s_{i_k}\}$ of M_1, the input symbol x takes this state to the union of the sets of next states for the elements of this set, that is, the union of the sets $f(s_{i_1}), f(s_{i_2}), \ldots, f(s_{i_k})$. The states of M_1 are all the subsets of S, the set of states of M_0, that are obtained in this way starting with s_0. (There are as

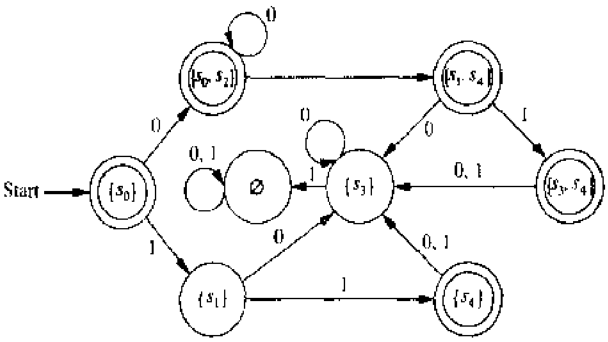


FIGURE 5 A Deterministic Automaton Equivalent to the Nondeterministic Automaton in Example 7.

many as 2^n states in the deterministic machine, where n is the number of states in the nondeterministic machine, since all subsets may occur as states, including the empty set, although usually far fewer states occur.) The final states of M_1 are those sets that contain a final state of M.

Suppose that an input string is recognized by M_0. Then one of the states that can be reached from s_0 using this input string is a final state (the reader should provide an inductive proof of this). This means that in M_1, this input string leads from $\{s_0\}$ to a set of states of M_0 that contains a final state. This subset is a final state of M_1, so this string is also recognized by M_1. Also, an input string not recognized by M_0 does not lead to any final states in M_0. (The reader should provide the details that prove this statement.) Consequently, this input string does not lead from $\{s_0\}$ to a final state in M_1. □

EXAMPLE 9 Find a deterministic finite-state automaton that recognizes the same language as the nondeterministic finite-state automaton in Example 7.

Solution: The deterministic automaton shown in Figure 5 is constructed from the nondeterministic automaton in Example 7. The states of this deterministic automaton are subsets of the set of all states of the nondeterministic machine. The next state of a subset under an input symbol is the subset containing the next states in the nondeterministic machine of all elements in this subset. For instance, on input of 0, $\{s_0\}$ goes to $\{s_0, s_2\}$, since s_0 has transitions to itself and to s_2 in the nondeterministic machine; the set $\{s_0, s_2\}$ goes to $\{s_1, s_4\}$ on input of 1, since s_0 goes just to s_1 and s_2 goes just to s_4 on input of 1 in the nondeterministic machine; and the set $\{s_1, s_4\}$ goes to $\{s_3\}$ on input of 0, since s_1 and s_4 both go to just s_3 on input of 0 in the deterministic machine. All subsets that are obtained in this way are included in the deterministic finite-state machine. Note that the empty set is one of the states of this machine, since it is the subset containing all the next states of $\{s_3\}$ on input of 1. The start state is $\{s_0\}$, and the set of final states are all those that include s_0 or s_4. ■

Exercises

1. Let $A = \{0, 11\}$ and $B = \{00, 01\}$. Find each of the following sets.

a) AB **b)** BA **c)** A^2 **d)** B^3

2. Show that if A is a set of strings, then $A\emptyset = \emptyset A = \emptyset$.

3. Find all pairs of sets of strings A and B for which $AB = \{10, 111, 1010, 1000, 10111, 101000\}$.

4. Show that the following equalities hold.
  a) $\{\lambda\}^* = \{\lambda\}$
  b) $(A^*)^* = A^*$ for every set of strings A

5. Describe the elements of the set A^* for the following values of A.
  a) $\{10\}$ **b)** $\{111\}$ **c)** $\{0, 01\}$ **d)** $\{1, 101\}$

6. Let V be an alphabet, and let A and B be subsets of V^*. Show that $|AB| \le |A||B|$.

7. Let V be an alphabet, and let A and B be subsets of V^* with $A \subseteq B$. Show that $A^* \subseteq B^*$.

8. Suppose that A is a subset of V^* where V is an alphabet. Prove or disprove each of the following statements.
  a) $A \subseteq A^2$ **b)** if $A = A^2$, then $\lambda \in A$ **c)** $A\{\lambda\} = A$
  d) $(A^*)^* = A^*$ **e)** $A^*A = A^*$ **f)** $|A^n| = |A|^n$

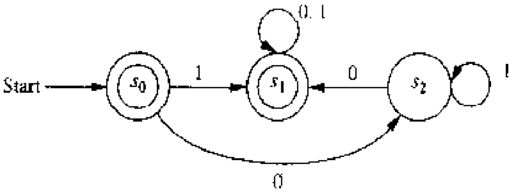
9. Determine whether the string 11101 is in each of the following sets.
  a) $\{0, 1\}^*$ **b)** $\{1\}^*\{0\}^*\{1\}^*$ **c)** $\{11\}\{1\}^*\{01\}$
  d) $\{11\}^*\{01\}^*$ **e)** $\{111\}^*\{0\}^*\{1\}$ **f)** $\{111, 000\}\{00, 01\}$

10. Determine whether each of the following strings is recognized by the deterministic finite-state automaton in Figure 1.
  a) 010 **b)** 1101 **c)** 1111110 **d)** 010101010

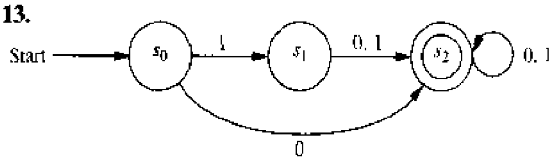
11. Determine whether all the strings in each of the following sets are recognized by the deterministic finite-state automaton in Figure 1.
  a) $\{0\}^*$ **b)** $\{0\}\{0\}^*$ **c)** $\{1\}\{0\}^*$
  d) $\{01\}^*$ **e)** $\{0\}^*\{1\}^*$ **f)** $\{1\}\{0, 1\}^*$

In Exercises 12–16 find the language recognized by the given deterministic finite-state automaton.

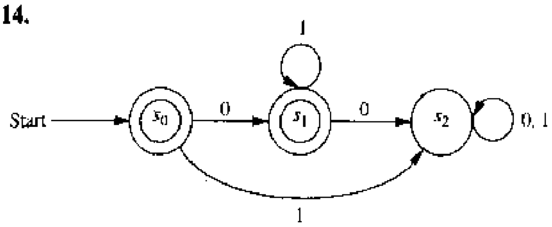
12.



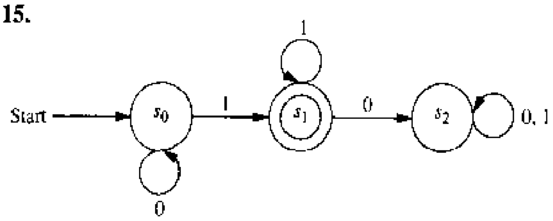
13.



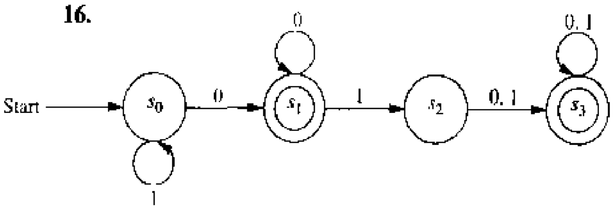
14.



15.

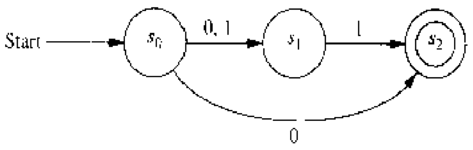


16.

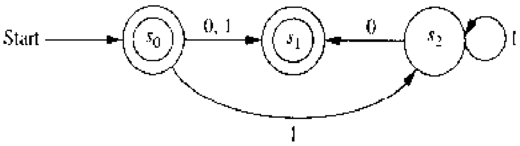


In Exercises 17–21 find the language recognized by the given nondeterministic finite-state automaton.

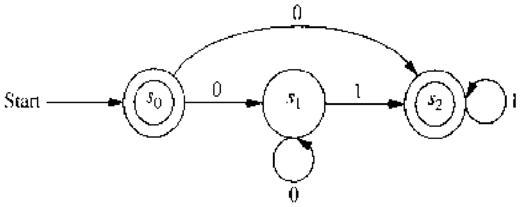
17.



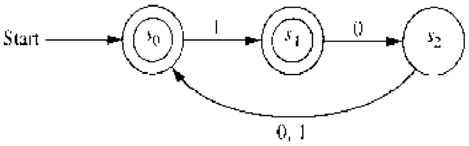
18.



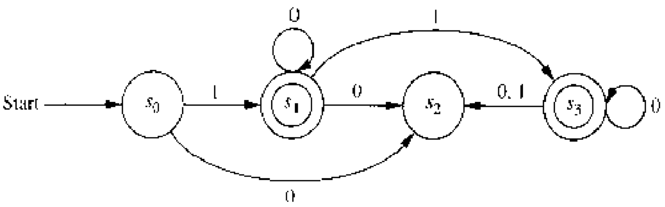
19.



20.



21.



22. Find a deterministic finite-state automaton that recognizes the same language as the nondeterministic finite-state automaton in Exercise 17.

23. Find a deterministic finite-state automaton that recognizes the same language as the nondeterministic finite-state automaton in Exercise 18.

24. Find a deterministic finite-state automaton that recognizes the same language as the nondeterministic finite-state automaton in Exercise 19.

25. Find a deterministic finite-state automaton that recognizes the same language as the nondeterministic finite-state automaton in Exercise 20.

26. Find a deterministic finite-state automaton that recognizes the same language as the nondeterministic finite-state automaton in Exercise 21.

27. Find a deterministic finite-state automaton that recognizes each of the following sets.

a) $\{0\}$ **b)** $\{1, 00\}$

c) $\{1^n \mid n = 2, 3, 4, \dots\}$

28. Find a nondeterministic finite-state automaton that recognizes each of the languages in Exercise 27, and has fewer states, if possible, than the deterministic automaton you found in that exercise.

***29.** Show that there is no finite-state automaton that recognizes the set of bit strings containing an equal number of 0s and 1s.

10.4

Language Recognition

INTRODUCTION

We have seen that finite-state automata can be used as language recognizers. What sets can be recognized by these machines? Although this seems like an extremely difficult problem, there is a simple characterization of the sets that can be recognized by finite state automata. This problem was first solved in 1956 by the American mathematician Stephen Kleene. He showed that there is a finite-state automaton which recognizes a set if and only if this set can be built up from the null set, the empty string, and singleton strings by taking concatenations, unions, and Kleene closures, in arbitrary order. Sets that can be built up in this way are called **regular sets.**

Regular grammars were defined in Section 10.1. Because of the terminology used, it is not surprising that there is a connection between regular sets, which are the sets recognized by finite-state automata, and regular grammars. In particular, a set is regular if and only if it is generated by a regular grammar.

Finally, there are sets that cannot be recognized by any finite-state automata. We will give an example of such a set. We will briefly discuss more powerful models of computation, such as pushdown automata and Turing machines, at the end of this section.

REGULAR SETS

The regular sets are those that can be formed using the operations of concatenation, union, and Kleene closure in arbitrary order, starting with the empty set, the empty string, and singleton sets. We will see that the regular sets are those which can be recognized using a finite-state automaton. To define regular sets we first need to define regular expressions.

DEFINITION 1. The *regular expressions* over a set I are defined recursively by:

the symbol $\emptyset$ is a regular expression;
the symbol λ is a regular expression;
the symbol x is a regular expression whenever $x \in I$;
the symbols $(\mathbf{AB})$, $(\mathbf{A} \cup \mathbf{B})$, and $\mathbf{A}^*$ are regular expressions whenever $\mathbf{A}$ and $\mathbf{B}$ are regular expressions.

Each regular expression represents a set specified by the following rules:

$\emptyset$ represents the empty set, that is, the set with no strings;
λ represents the set $\{\lambda\}$, which is the set containing the empty string;
x represents the set $\{x\}$ containing the string with one symbol x;
$(\mathbf{AB})$ represents the concatenation of the sets represented by $\mathbf{A}$ and by $\mathbf{B}$;
$(\mathbf{A} \cup \mathbf{B})$ represents the union of the sets represented by $\mathbf{A}$ and by $\mathbf{B}$;
$\mathbf{A}^*$ represents the Kleene closure of the set represented by $\mathbf{A}$.

Sets represented by regular expressions are called **regular sets.** Henceforth regular expressions will be used to describe regular sets, so when we refer to the regular set **A**, we will mean the regular set represented by the regular expression **A**. The following example shows how regular expressions are used to specify regular sets.

EXAMPLE 1 What are the strings in the regular sets specified by the regular expressions **10***, **(10)***, **0 ∪ 01**, **0(0 ∪ 1)***, and **(0*1)***?

Solution: The regular sets represented by these expressions are given in Table 1, as the reader should verify. ■

KLEENE'S THEOREM

In 1956 Kleene proved that regular sets are the sets which are recognized by a finite-state automaton. Consequently, this important result is called Kleene's theorem.

THEOREM 1 KLEENE'S THEOREM **A set is regular if and only if it is recognized by a finite-state automaton.**

Kleene's theorem is one of the central results in automata theory. We will prove the *only if* part of this theorem, namely, that every regular set is recognized by a finite-state automaton. The proof of the *if* part, that a set recognized by a finite-state automaton is regular, is left as an exercise for the reader.

Proof: Recall that a regular set is defined in terms of regular expressions, which are defined recursively. We can prove that every regular set is recognized by a finite-state automaton if we can do the following things.

1. Show that ∅ is recognized by a finite-state automaton.
2. Show that $\{\lambda\}$ is recognized by a finite-state automaton.
3. Show that $\{a\}$ is recognized by a finite-state automaton whenever a is a symbol in I.
4. Show that AB is recognized by a finite-state automaton whenever both A and B are.
5. Show that $A \cup B$ is recognized by a finite-state automaton whenever both A and B are.
6. Show that A^* is recognized by a finite-state automaton whenever A is. □

TABLE 1

Expression	*Strings*
10*	A 1 followed by any number of 0s (including no zeros)
(10)*	Any number of copies of 10 (including the null string)
0 ∪ 01	The string 0 or the string 01
0(0 ∪ 1)*	Any string beginning with 0
(0*1)*	Any string not ending with 0

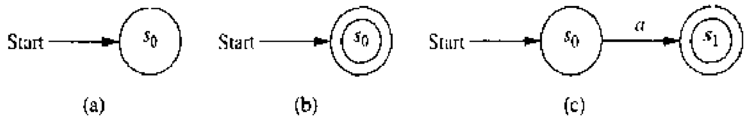


FIGURE 1 **Nondeterministic Finite-State Automata That Recognize Some Basic Sets.**

We now consider each of these tasks. First, we show that $\varnothing$ is recognized by a nondeterministic finite-state automaton. To do this, all we need is an automaton with no final states. Such an automaton is shown in Figure 1(a).

Second, we show that $\{\lambda\}$ is recognized by a finite-state automaton. To do this, all we need is an automaton that recognizes λ, the null string, but not any other string. This can be done by making the start state s_0 a final state and having no transitions so that no other string takes s_0 to a final state. The nondeterministic automaton in Figure 1(b) shows such a machine.

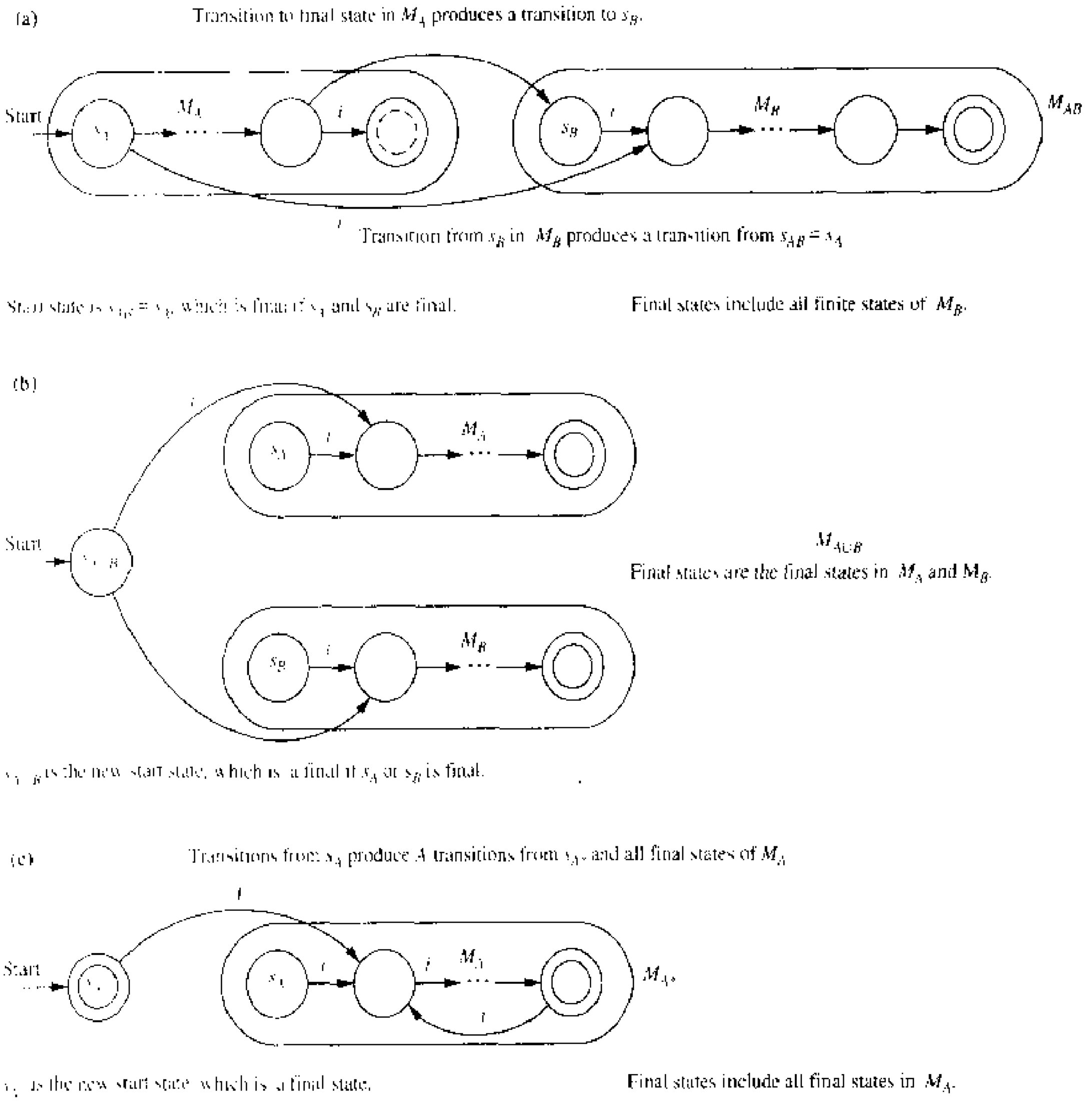
Third, we show that $\{a\}$ is recognized by a nondeterministic finite-state automaton. To do this, we can use a machine with a starting state s_0 and a final state s_1. We have a transition from s_0 to s_1 when the input is a, and no other transitions. The only string recognized by this machine is a. This machine is shown in Figure 1(c).

Next, we show that AB and $A \cup B$ can be recognized by finite-state automata if A and B are languages recognized by finite-state automata. Suppose that A is recognized by $M_A = (S_A, I, f_A, s_A, F_A)$ and B is recognized by $M_B = (S_B, I, f_B, s_B, F_B)$.

We begin by constructing a finite-state machine $M_{AB} = (S_{AB}, I, f_{AB}, s_{AB}, F_{AB})$ that recognizes AB, the concatenation of A and B. We build such a machine by combining the machines for A and B in series, so a string in A takes the combined machine from s_A, the start state of M_A, to s_B, the start state of M_B. A string in B should take the combined machine from s_B to a final state of the combined machine. Consequently, we make the following construction. Let S_{AB} be $S_A \cup S_B$. The starting state s_{AB} is the same as s_A. The set of final states, F_{AB}, is the set of final states of M_B with s_{AB} included if and only if $\lambda \in A \cap B$. The transitions in M_{AB} include all transitions in M_A and in M_B, as well as some new transitions. For every transition in M_A that leads to a final state, we form a transition in M_{AB} from the same state to s_B, on the same input. In this way, a string in A takes M_{AB} from s_{AB} to s_B, and then a string in B takes s_B to a final state of M_{AB}. Moreover, for every transition from s_B we form a transition in M_{AB} from s_{AB} to the same state. Figure 2(a) contains an illustration of this construction.

We now construct a machine $M_{A\cup B} = \{S_{A\cup B}, I, f_{A\cup B}, s_{A\cup B}, F_{A\cup B}\}$ that recognizes $A \cup B$. This automaton can be constructed by combining M_A and M_B in parallel, using a new start state that has the transitions that both s_A and s_B have. Let $S_{A\cup B} = S_A \cup S_B \cup \{s_{A\cup B}\}$, where $s_{A\cup B}$ is a new state that is the start state of $M_{A\cup B}$. Let the set of final states $F_{A\cup B}$ be $F_A \cup F_B \cup \{s_{A\cup B}\}$ if $\lambda \in A \cup B$, and $F_A \cup F_B$ otherwise. The transitions in $M_{A\cup B}$ include all those in M_A and in M_B. Also, for each transition from s_A to a state s on input i we include a transition from $s_{A\cup B}$ to s on input i, and for each transition from s_B to a state s on input i we include a transition from $s_{A\cup B}$ to s on input i. In this way, a string in A leads from $s_{A\cup B}$ to a final state in the new machine, and a string in B leads from $s_{A\cup B}$ to a final state in the new machine. Figure 2(b) illustrates the construction of $M_{A\cup B}$.

Finally, we construct $M_{A^*} = (S_{A^*}, I, f_{A^*}, s_{A^*}, F_{A^*})$, a machine that recognizes A^*, the Kleene closure of A. Let S_{A^*} include all states in S_A and one additional state s_{A^*}, which is the starting state for the new machine. The set of final states F_{A^*} includes all states in F_A as well as the start state s_{A^*}, since λ must be recognized. To recognize



Transition to final state in M_A produces a transition to s_B.

Transition from s_B in M_B produces a transition from $s_{AB} = s_A$

Start state is $s_{AB} = s_A$, which is final if s_A and s_B are final.

Final states include all finite states of M_B.

$s_{A\cup B}$ is the new start state, which is a final if s_A or s_B is final.

Final states are the final states in M_A and M_B.

Transitions from s_A produce A transitions from s_{A^*} and all final states of M_A

s_{A^*} is the new start state, which is a final state.

Final states include all final states in M_A.

FIGURE 2 **Building Automata to Recognize Concatenations, Unions, and Kleene Closures.**

concatenations of arbitrarily many strings from A, we include all the transitions in M_A, as well as transitions from s_{A^*} that match the transitions from s_A, and transitions from each final state that match the transitions from s_A. With this set of transitions, a string made up of concatenations of strings from A will take s_{A^*} to a final state when the first string in A has been read, returning to a final state when the second string in A has been read, and so on. Figure 2(c) illustrates the construction we used.

A nondeterministic finite-state automaton can be constructed for any regular set using the procedure described in this proof. We illustrate how this is done with the following example.

EXAMPLE 2 Construct a nondeterministic finite-state automaton that recognizes the regular set $1^* \cup 01$.

Solution: We begin by building a machine that recognizes $\mathbf{1}^*$. This is done using the machine that recognizes $\mathbf{1}$ and then using the construction for M_{A^*} described in the proof. Next, we build a machine that recognizes $\mathbf{01}$, using machines that recognize $\mathbf{0}$ and $\mathbf{1}$ and the construction in the proof for M_{AB}. Finally, using the construction in the proof for $M_{A \cup B}$, we construct the machine for $\mathbf{1}^* \cup \mathbf{01}$. The finite-state automata used in this construction are shown in Figure 3. The states in the successive machines have been labeled using different subscripts, even when a state is formed from one previously used in another machine. Note that the construction given here does not produce the simplest machine that recognizes $\mathbf{1}^* \cup \mathbf{01}$. A much simpler machine that recognizes this set is shown in Figure 3(b). ■

REGULAR SETS AND REGULAR GRAMMARS

In Section 10.1 we introduced phrase-structure grammars and defined different types of grammars. In particular we defined regular, or type 3, grammars, which are grammars of the form $G = (V, T, S, P)$ where each production is of the form $S \rightarrow \lambda$, $A \rightarrow a$, or $A \rightarrow aB$, where a is a terminal symbol, and A and B are nonterminal symbols. As the terminology suggests, there is a close connection between regular grammars and regular sets.

THEOREM 2 **A set is generated by a regular grammar if and only if it is a regular set.**

Proof: First we show that a set generated by a regular grammar is a regular set. Suppose that $G = (V, T, S, P)$ is a regular grammar generating the set $L(G)$. To show that $L(G)$ is regular we will build a nondeterministic finite-state machine $M = (S, I, f, s_0, F)$ that recognizes $L(G)$. Let S, the set of states, contain a state s_A for each nonterminal symbol A of G and an additional state s_F, which is a final state. The start state s_0 is the state formed from the start symbol S. The transitions of M are formed from the productions of G in the following way. A transition from s_A to s_F on input of a is included if $A \rightarrow a$ is a production, and a transition from s_A to s_B on input of a is included if $A \rightarrow aB$ is a production. The set of final states includes s_F and also includes s_0 if $S \rightarrow \lambda$ is a production in G. It is not hard to show that the language recognized by M equals the language generated by the grammar G, that is, the $L(M) = L(G)$. This can be done by determining the words that lead to a final state. The details are left as an exercise for the reader. □

Before giving the proof of the converse, we illustrate how a nondeterministic machine is constructed that recognizes the same set as a regular grammar.

EXAMPLE 3 Construct a nondeterministic finite-state automaton that recognizes the language generated by the regular grammar $G = (V, T, S, P)$, where $V = \{0, 1, A, S\}$, $T = \{0, 1\}$, and the productions in P are $S \rightarrow 1A$, $S \rightarrow 0$, $S \rightarrow \lambda$, $A \rightarrow 0A$, $A \rightarrow 1A$, and $A \rightarrow 1$.

Solution: The state diagram for a nondeterministic finite-state automaton that recognizes $L(G)$ is shown in Figure 4. This automaton is constructed following the procedure described in the proof. In this automaton s_0 is the state corresponding to S, s_1 is the state corresponding to A, and s_2 is the final state. ■

We now complete the proof of Theorem 2.

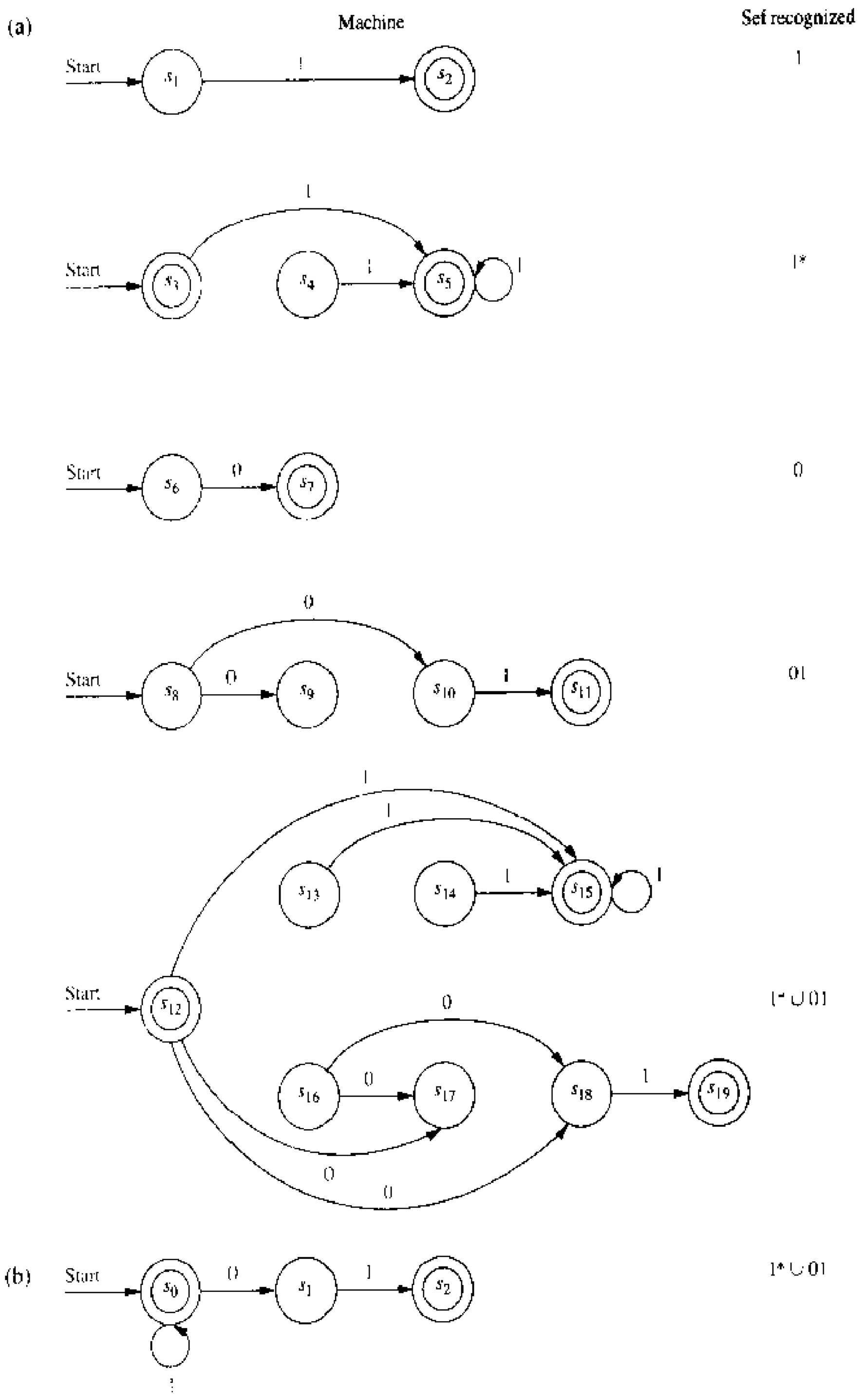

(a)
Machine
Set recognized
Start
s_1
1
s_2
1
Start
s_3
s_4
s_5
1
1
1
1*
Start
s_6
0
s_7
0
Start
s_8
0
s_9
0
s_{10}
1
s_{11}
01
s_{13}
s_{14}
s_{15}
1
1
1
1
Start
s_{12}
0
s_{16}
0
s_{17}
s_{18}
1
s_{19}
0
0
1* ∪ 01
(b)
Start
s_0
0
s_1
1
s_2
1
1* ∪ 01

FIGURE 3 Nondeterministic Finite-State Automata Recognizing $1^* \cup 01$.

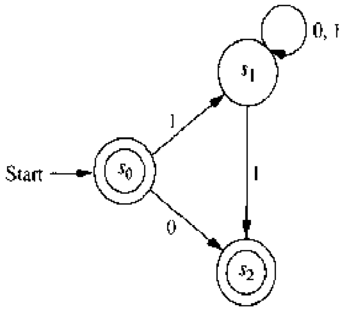


FIGURE 4 **A Nondeterministic Finite-State Automaton Recognizing $L(G)$.**

Proof: We now show that if a set is regular, then there is a regular grammar that generates it. Suppose that M is a finite-state machine which recognizes this set with the property that s_0, the starting state of M, is never the next state for a transition. (We can find such a machine by Exercise 14.) The language $G = (V, T, S, P)$ is defined as follows. The set V of symbols of G is formed by assigning a symbol to each state of S and each input symbol in I. The set T of terminal symbols of G is the symbols of G formed from the input symbols in I. The start symbol S is the symbol formed from the start state s_0. The set P of productions in G is formed from the transitions in M. In particular, if the state s goes to a final state under input a, then the production $A_s \rightarrow a$ is included in P, where A_s is the nonterminal symbol formed from the state s. If the state s goes to the state t on input a, then the production $A_s \rightarrow aA_t$ is included in P. The production $S \rightarrow \lambda$ is included in P if and only if $\lambda \in L(M)$. Since the productions of G correspond to the transitions of M and the productions leading to terminals correspond to transitions to final states, it is not hard to show that $L(G) = L(M)$. We leave the details as an exercise for the reader. ☐

The following example illustrates the construction used to produce a grammar from an automaton that generates the language recognized by this automaton.

EXAMPLE 4 Find a regular grammar that generates the regular set recognized by the finite-state automaton shown in Figure 5.

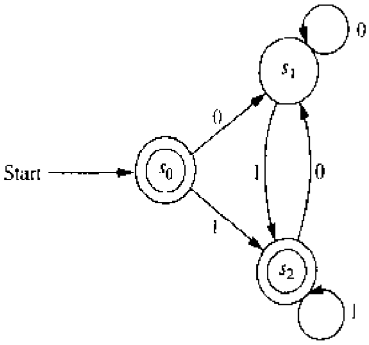


FIGURE 5 **A Finite-State Automaton.**

Solution: The grammar $G = (V, T, S, P)$ generates the set recognized by this automaton where $G = \{S, A, B, 0, 1\}$; where the symbols S, A, and B correspond to the states s_0, s_1, and s_2, respectively; $T = \{0, 1\}$; S is the start symbol; and the productions are $S \to 0A, S \to 1B, S \to 1, S \to \lambda, A \to 0A, A \to 1B, A \to 1, B \to 0A, B \to 1B$, and $B \to 1$. ■

A SET NOT RECOGNIZED BY A FINITE-STATE AUTOMATON

We have seen that a set is recognized by a finite-state automaton if and only if it is regular. We will now show that there are sets which are not regular by describing one such set. The technique used to show that this set is not regular illustrates an important method for showing that certain sets are not regular.

EXAMPLE 5 Show that the set $\{0^n 1^n \mid n = 0, 1, 2, \ldots\}$, made up of all strings consisting of a block of 0s followed by a block of an equal number of 1s, is not regular.

Solution: Suppose that this set were regular. Then there would be a deterministic finite-state automaton $M = (S, I, f, s_0, F)$ recognizing it. Let N be the number of states in this machine, that is, $N = |S|$. Since M recognizes all strings made up of a number of 0s followed by an equal number of 1s, M must recognize $0^N 1^N$. Let $s_0, s_1, s_2, \ldots, s_{2N}$ be the sequence of states which is obtained starting at s_0 and using the symbols of $0^N 1^N$ as input so that $s_1 = f(s_0, 0)$, $s_2 = f(s_1, 0), \ldots, s_N = f(s_{N-1}, 0)$, $s_{N+1} = f(s_N, 1), \ldots, s_{2N} = f(s_{2N-1}, 1)$. Note that s_{2N} is a final state.

Since there are only N states, the pigeonhole principle shows that at least two of the first $N + 1$ of the states, which are $s_0, \ldots, s_N$, must be the same. Say that s_i and s_j are two such identical states, with $0 \leq i < j \leq N$. This means that $f(s_i, 0^t) = s_j$ where $t = i - j$. It follows that there is a loop leading from s_i back to itself, obtained using the input 0 a total of t times, in the state diagram shown in Figure 6.

Now consider the input string $0^N 0^t 1^N = 0^{N+t} 1^N$. There are t more consecutive 0s at the start of this block than there are consecutive 1s that follow it. Since this string is not of the form $0^n 1^n$ (since it has more 0s that 1s), it is not recognized by M. Consequently, $f(s_0, 0^{N+t} 1^N)$ cannot be a final state. However, when we use the string $0^{N+t} 1^N$

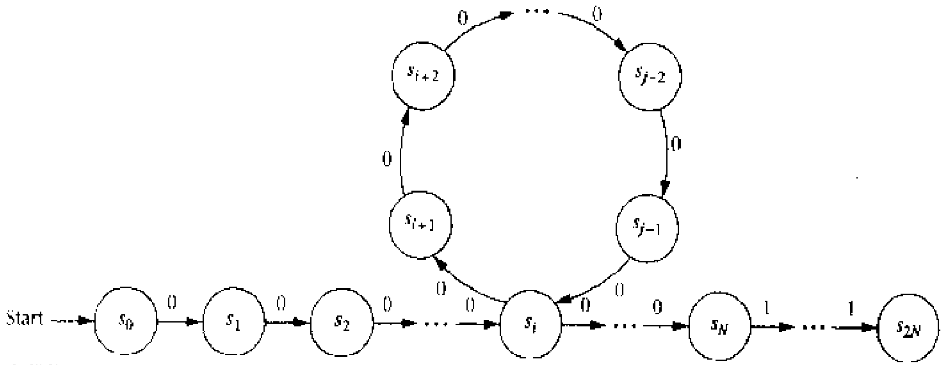


FIGURE 6 The Path Produced by $0^N 1^N$.

as input, we end up in the same state as before, namely, s_{2N}. The reason for this is that the extra t 0s in this string take us around the loop from s_i back to itself an extra time, as shown in Figure 6. Then the rest of the string leads us to exactly the same state as before. This contradiction shows that $\{0^n1^n \mid n = 1, 2, \ldots\}$ is not regular. ■

MORE POWERFUL TYPES OF MACHINES

Finite-state automata are unable to carry out many computations. The main limitation of these machines is their finite amount of memory. This prevents them from recognizing languages that are not regular, such as $\{0^n1^n \mid n = 0, 1, 2, \ldots\}$. Since a set is regular if and only if it is the language generated by a regular grammar, Example 5 shows that there is no regular grammar which generates the set $\{0^n1^n \mid n = 0, 1, 2, \ldots\}$. However, there is a context-free grammar which recognizes this set. Such a grammar was given in Example 5 in Section 10.1.

Because of the limitations of finite-state machines, it is necessary to use other, more powerful, models of computation. One such model is the **pushdown automaton.** A pushdown automaton includes everything in a finite-state automaton, as well as a stack, which provides unlimited memory. Symbols can be placed on the top or taken off the top of the stack. A set is recognized in one of two ways by a pushdown automaton. First, a set is recognized if the set consists of all the strings that produce an empty stack when they are used as input. Second, a set is recognized if it consists of all the strings that lead to a final state when used as input. It can be shown that a set is recognized by a pushdown automaton if and only if it is the language generated by a context-free grammar.

However, there are sets that cannot be expressed as the language generated by a context-free grammar. One such set is $\{0^n1^n2^n \mid n = 0, 1, 2, \ldots\}$. We will indicate why this set cannot be recognized by a pushdown automaton, but we will not give a proof, since we have not developed the machinery needed. (However, one method of proof is given in Exercise 28 of the supplementary exercises at the end of this chapter.) The stack can be used to show that a string begins with a sequence of 0s followed by an equal number of 1s by placing a symbol on the stack for each 0 (as long as only 0s are read), and removing one of these symbols for each 1 (as long as only 1s following the 0s are read). But once this is done, the stack is empty, and there is no way to determine that there are the same number of 2s in the string as 0s.

There are other machines called **linear bounded automata,** more powerful than pushdown automata, that can recognize sets such as $\{0^n1^n2^n \mid n = 0, 1, 2, \ldots\}$. In particular, linear bounded automata can recognize context-sensitive languages. However, these machines cannot recognize all the languages generated by phrase-structure grammars. To avoid the limitations of special types of machines, the model known as a **Turing machine,** named after the British mathematician Alan Turing, is used. A Turing machine is made up of everything included in a finite-state machine together with a tape, which is infinite in both directions. A Turing machine has read and write capabilities on the tape, and it can move back and forth along this tape. Turing machines can recognize all languages generated by phrase-structure grammars. In addition, Turing machines can model all the computations that can be performed on a computing machine. Because of their power, Turing machines are extensively studied in theoretical computer science. We will briefly study them in the next section.

Exercises

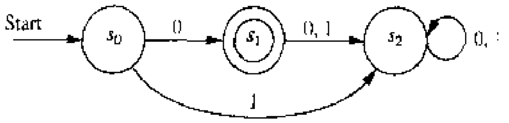
1. Describe in words the strings in each of the following regular sets.
   a) $\mathbf{1^*0}$ b) $\mathbf{1^*00^*}$
   c) $\mathbf{111 \cup 001}$ d) $\mathbf{(1 \cup 00)^*}$
   e) $\mathbf{(00^*1)^*}$ f) $\mathbf{(0 \cup 1)(0 \cup 1)^*00}$
2. Determine whether 1011 belongs to each of the following regular sets.
   a) $\mathbf{10^*1^*}$ b) $\mathbf{0^*(10 \cup 11)^*}$
   c) $\mathbf{1(01)^*1^*}$ d) $\mathbf{1^*01(0 \cup 1)}$
   e) $\mathbf{(10)^*(11)^*}$ f) $\mathbf{1(00)^*(11)^*}$
   g) $\mathbf{(10)^*1011}$ h) $\mathbf{(1 \cup 00)(01 \cup 0)1^*}$
3. Express each of the following sets using a regular expression.
   a) the set of strings of one or more 0s followed by a 1
   b) the set of strings of two or more symbols followed by three or more 0s
   c) the set of strings with either no 1 preceding a 0 or no 0 preceding a 1
   d) the set of strings containing a string of 1s so that the number of 1s equals 2 modulo 3, followed by an even number of 0s
4. Construct deterministic finite-state automata that recognize the following sets from I^*, where I is an alphabet.
   a) $\emptyset$ b) $\{\lambda\}$ c) $\{a\}$, where $a \in I$

*5. Show that if A is a regular set, then A^R, the set of all reversals of strings in A, is also regular.

6. Find a finite-state automaton that recognizes
   a) $\{\lambda, 0\}$. b) $\{0, 11\}$. c) $\{0, 11, 000\}$.
7. Using the constructions described in the proof of Kleene's theorem, find nondeterministic finite-state automata that recognize each of the following sets.
   a) $\mathbf{0^*1^*}$ b) $\mathbf{(0 \cup 11)^*}$ c) $\mathbf{01^* \cup 00^*1}$
8. Construct a nondeterministic finite-state automaton that recognizes the language generated by the regular grammar $G = (V, T, S, P)$ where $V = \{0, 1, S, A, B\}$, $T = \{0, 1\}$, S is the start symbol, and the set of productions is
   a) $S \to 0A$, $S \to 1B$, $A \to 0$, $B \to 0$.
   b) $S \to 1A$, $S \to 0$, $S \to \lambda$, $A \to 0B$, $B \to 1B$, $B \to 1$.
   c) $S \to 1B$, $S \to 0$, $A \to 1A$, $A \to 0B$, $A \to 1$, $A \to 0$, $B \to 1$.

In Exercises 9–11 construct a regular grammar $G = (V, T, S, P)$ that generates the language recognized by the given finite-state machine.

9.



---

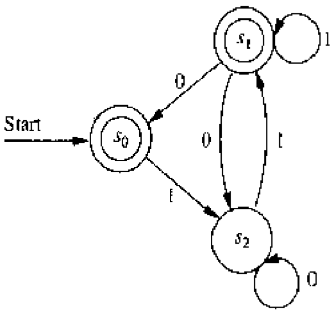
web

Alan Mathison Turing (1912–1954). Alan Turing was born in London, although he was conceived in India, where his father was employed in the Indian Civil Service. As a boy, he was fascinated by chemistry, performing a wide variety of experiments, and by machinery. Turing attended Sherborne, an English boarding school. In 1931 he won a scholarship to King's College, Cambridge. After completing his dissertation, which included a rediscovery of the central limit theorem, a famous theorem in statistics, he was elected a fellow of his college. In 1935 Turing became fascinated with the decision problem, a problem posed by the great German mathematician Hilbert, which asked whether there is a general method that can be applied to any assertion to determine whether the assertion is true. Turing enjoyed running (later in life running as a serious amateur in competitions), and one day, while resting after a run, he discovered the key ideas needed to solve the decision problem. In his solution, he invented what is now called a **Turing machine** as the most general model of a computing machine. Using these machines, he found a problem, involving what he called computable numbers, that could not be decided using a general method.

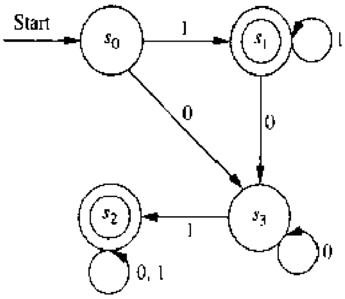
From 1936 to 1938 Turing visited Princeton University to work with Alonzo Church, who had also solved Hilbert's decision problem. In 1939 Turing returned to King's College. However, at the outbreak of World War II, he joined the Foreign Office, performing cryptanalysis of German ciphers. His contribution to the breaking the code of the Enigma, a mechanical German cipher machine, played an important role in winning the war.

After the war, Turing worked on the development of early computers. He was interested in the ability of machines to think, proposing that if a computer could not be distinguished from a person based on written replies to questions, it should be considered to be "thinking." He was also interested in biology, having written on morphogenesis, the development of form in organisms. In 1954 Turing committed suicide by taking cyanide, without leaving a clear explanation. Legal troubles related to a homosexual relationship and hormonal treatments mandated by the court to lessen his sex drive may have been factors in his decision to end his life.

10.



11.



12. Show that the finite-state automaton constructed from a regular grammar in the proof of Theorem 2 recognizes the set generated by this grammar.

13. Show that the regular grammar constructed from a finite-state automaton in the proof of Theorem 2 generates the set recognized by this automaton.

14. Show that every nondeterministic finite-state automaton is equivalent to another such automaton that has the property that its starting state is never revisited.

***15.** Let $M = (S, I, f, s_0, F)$ be a deterministic finite-state automaton. Show that the language recognized by M, $L(M)$, is infinite if and only if there is a word x recognized by M with $l(x) \geq |S|$.

***16.** One important technique used to prove that certain sets are not regular is the **pumping lemma.** The pumping lemma states that if $M = (S, I, f, s_0, F)$ is a deterministic finite-state automaton and if x is a string in $L(M)$, the language recognized by M, with $l(x) \geq |S|$, then there are strings u, v, and w in I^* such that $x = uvw$, $l(uv) \leq |S|$ and $l(v) \geq 1$, and $uv^iw \in L(M)$ for $i = 0, 1, 2, \ldots$. Prove the pumping lemma. (*Hint:* Use the same idea as was used in Example 5.)

***17.** Show that the set $\{0^{2n}1^n\}$ is not regular. You may use the pumping lemma given in Exercise 16.

***18.** Show that the set $\{1^{n^2} \mid n = 0, 1, 2, \ldots\}$ is not regular. You may use the pumping lemma given in Exercise 16.

***19.** Show that the set of palindromes over $\{0, 1\}$ is not regular. You may use the pumping lemma given in Exercise 16. (*Hint:* Consider strings of the form 0^N10^N.)

****20.** Show that a set recognized by a finite-state automaton is regular. (This is the *if* part of Kleene's theorem.)

10.5

Turing Machines

INTRODUCTION

The finite-state automata studied earlier in this chapter cannot be used as general models of computation. They are limited in what they can do. For example, finite-state automata are able to recognize regular sets, but not able to recognize many easy-to-describe sets, including $\{0^n1^n \mid n \geq 0\}$, which computers recognize using memory. We can use finite-state automata to compute relatively simple functions such as the sum of two numbers, but we cannot use them to compute functions that computers can, such as the product of two numbers. To overcome these deficiencies we can use a more powerful type of machine known as a Turing machine, after Alan Turing, the famous mathematician and computer scientist who invented them in the 1930s.

Basically, a Turing machine consists of a control unit, which at any step is in one of finitely many different states, together with a tape divided into cells, which is infinite in both directions. Turing machines have read and write capabilities on the tape as the control unit moves back and forth along this tape, changing states depending on the tape symbol read. Turing machines are more powerful than finite-state machines because they include memory capabilities that finite-state machines lack. We will show how to use Turing machines to recognize sets, including sets that cannot be recognized by finite-state machines. We will also show how to compute functions using Turing machines. Turing machines are the most general models of computation; essentially, they can do whatever a computer can do.

DEFINITION OF TURING MACHINES

We now give the formal definition of a Turing machine. Afterward we will explain how this formal definition can be interpreted in terms of a control head which can read and write symbols on a tape and move either right or left along the tape.

DEFINITION 1. A *Turing machine* $T = (S, I, f, s_0)$ consists of a finite set S of states, an alphabet I containing the blank symbol B, a partial function f from $S \times I$ to $S \times I \times \{R, L\}$, and a starting state s_0.

Recall from the preamble to Exercise 39 in Section 1.6 that a partial function is defined only for those elements in its domain of definition. This means that for some (state, symbol) pairs the partial function f may be undefined, but for a pair for which it is defined, there is a unique (state, symbol, direction) triple associated to this pair.

To interpret this definition in terms of a machine, consider a control unit and a tape divided into cells, infinite in both directions, having only a finite number of nonblank symbols on it at any given time, as pictured in Figure 1. The action of the Turing machine at each step of its operation depends on the value of the partial function f for the current state and tape symbol.

At each step, the control unit reads the current tape symbol x. If the control unit is in state s and if the partial function f is defined for the pair (s, x) with $f(s, x) = (s', x', d)$, the control unit:

1. enters the state s',
2. writes the symbol x' in the current cell, erasing x, and
3. moves right one cell if $d = R$ or moves left one cell if $d = L$.

We write this step as the five-tuple (s, x, s', x', d). If the partial function f is undefined for the pair (s, x), then the Turing machine T will *halt*.

A common way to define a Turing machine is to specify a set of five-tuples of the form (s, x, s', x', d). The set of states and input alphabet is implicitly defined when such a definition is used.

At the beginning of its operation a Turing machine is assumed to be in the initial state s_0 and to be positioned over the leftmost nonblank symbol on the tape. If the tape is all blank, the control head can be positioned over any cell. We will call the positioning of the control head over the leftmost nonblank tape symbol the *initial position* of the machine.

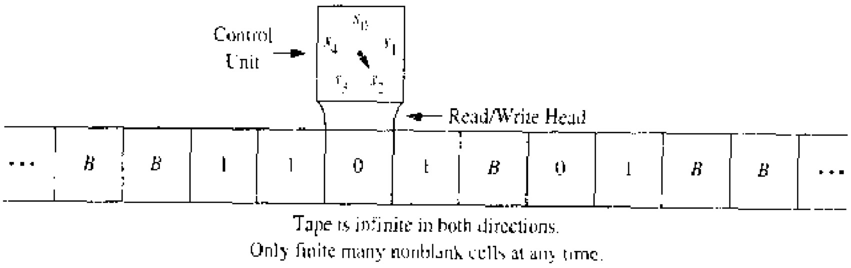

Control Unit
s_0
s_4
s_1
s_3
s_2
Read/Write Head
...
B
B
1
1
0
1
B
0
1
B
B
...
Tape is infinite in both directions.
Only finite many nonblank cells at any time.

FIGURE 1 **A Representation of a Turing Machine.**

The following example illustrates how a Turing machine works.

EXAMPLE 1 What is the final tape when the Turing machine T defined by the seven five-tuples $(s_0, 0, s_0, 0, R)$, $(s_0, 1, s_1, 1, R)$, (s_0, B, s_3, B, R), $(s_1, 0, s_0, 0, R)$, $(s_1, 1, s_2, 0, L)$, (s_1, B, s_3, B, R), $(s_2, 1, s_3, 0, R)$ is run on the tape shown in Figure 2(a)?

Solution: We start the operation with T in state s_0 and with T positioned over the leftmost nonblank symbol on the tape. The first step, using the five-tuple $(s_0, 0, s_0, 0, R)$, reads the 0 in the leftmost nonblank cell, stays in state s_0, writes a 0 in this cell, and moves one cell right. The second step, using the five-tuple $(s_0, 1, s_1, 1, R)$, reads the 1 in the current cell, enters state s_1, writes a 1 in this cell, and moves one cell right. The third step, using the five-tuple $(s_1, 0, s_0, 0, R)$, reads the 0 in the current cell, enters state

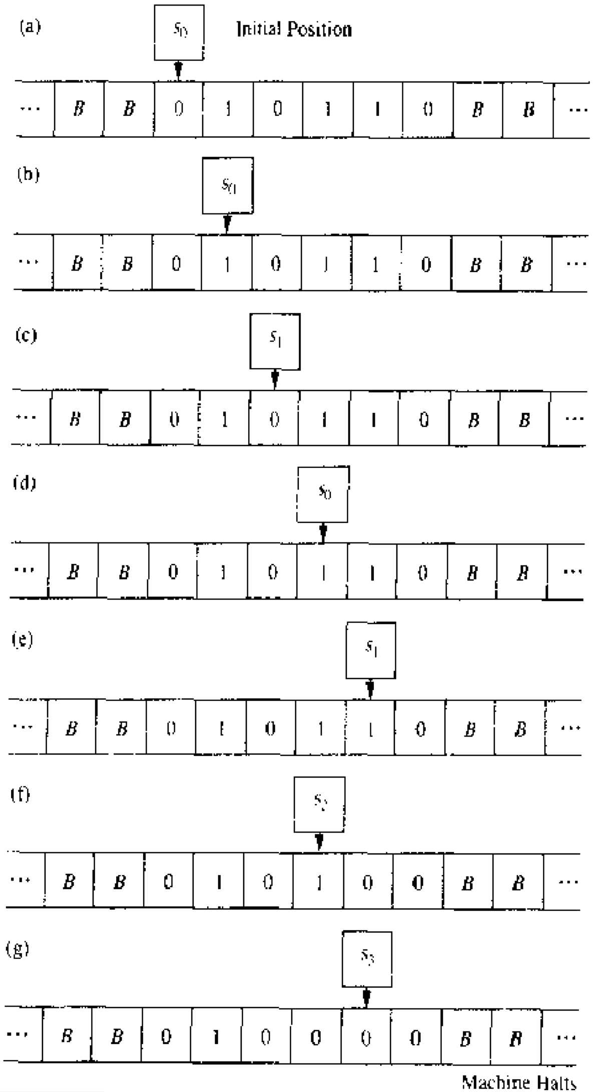

(a)
s_0
Initial Position
… B B 0 1 0 1 1 0 B B …
(b)
s_0
… B B 0 1 0 1 1 0 B B …
(c)
s_1
… B B 0 1 0 1 1 0 B B …
(d)
s_0
… B B 0 1 0 1 1 0 B B …
(e)
s_1
… B B 0 1 0 1 1 0 B B …
(f)
s_2
… B B 0 1 0 1 0 0 B B …
(g)
s_3
… B B 0 1 0 0 0 0 B B …
Machine Halts

FIGURE 2 **The Steps Produced by Running *T* on the Tape in Figure 1.**

s_0, writes a 0 in this cell, and moves one cell right. The fourth step, using the five-tuple $(s_0, 1, s_1, 1, R)$, reads the 1 in the current cell, enters state s_1, writes a 1 in this cell, and moves right one cell. The fifth step, using the five-tuple $(s_1, 1, s_2, 0, L)$, reads the 1 in the current cell, enters state s_2, writes a 0 in this cell, and moves left one cell. The sixth step, using the five-tuple $(s_2, 1, s_3, 0, R)$, reads the 1 in the current cell, enters the state s_3, writes a 0 in this cell, and moves right one cell. Finally, in the seventh step, the machine halts because there is no five-tuple beginning with the pair $(s_3, 0)$ in the description of the machine. The steps are shown in Figure 2.

Note that T changes the first pair of consecutive 1s on the tape to 0s and then halts. ■

USING TURING MACHINES TO RECOGNIZE SETS

Turing machines can be used to recognize sets. To do so requires that we define the concept of a final state as follows. A *final state* of a Turing machine T is a state that is not the first state in any five-tuple in the description of T using five-tuples (for example, state s_3 in Example 1).

We can now define what it means for a Turing machine to recognize a string. Given a string, we write consecutive symbols in this string in consecutive cells.

DEFINITION 2. **Let V be a subset of an alphabet T. A Turing machine $T = (S, I, f, s_0)$ recognizes a string x in V^* if and only if T, starting in the initial position when x is written on the tape, halts in a final state. T is said to recognize a subset A of V^* if x is recognized by T if and only if x belongs to A.**

Note that to recognize a subset A of V^* we can use symbols not in V. This means that the input alphabet I may include symbols not in V. These extra symbols are often used as markers (see Example 3).

When does a Turing machine T *not* recognize a string x in V^*? The answer is that x is not recognized if T does not halt or halts in a state that is not final when it operates on a tape containing the symbols of x in consecutive cells, starting in the initial position. (The reader should understand that this is one of many possible ways to define how to recognize sets using Turing machines.)

We illustrate this concept with the following example.

EXAMPLE 2 Find a Turing machine that recognizes the set of bit strings which have a 1 as their second bit (that is, the regular set $(\mathbf{0} \vee \mathbf{1})\mathbf{1}(\mathbf{0} \vee \mathbf{1})^*$).

Solution: We want a Turing machine that, starting at the leftmost nonblank tape cell, moves right, and determines whether the second symbol is a 1. If the second symbol is 1, the machine should move into a final state. If the second symbol is not a 1, the machine should not halt or it should halt in a nonfinal state.

To construct such a machine, we include the five-tuples $(s_0, 0, s_1, 0, R)$ and $(s_0, 1, s_1, 1, R)$ to read in the first symbol and put the Turing machine in state s_1. Next, we include the five-tuples $(s_1, 0, s_2, 0, R)$ and $(s_1, 1, s_3, 1, R)$ to read in the second symbol and either move to state s_2 if this symbol is a 0, or to state s_3 if this symbol is a 1. We do not want to recognize strings that have a 0 as their second bit, so s_2 should not be a final state. We want s_3 to be a final state. So, we can include the five-tuple $(s_2, 0, s_2, 0, R)$. Since we do not want to recognize the empty string nor a string with one bit, we also include the five-tuples $(s_0, B, s_2, 0, R)$ and $(s_1, B, s_2, 0, R)$.

The Turing machine T consisting of the seven five-tuples listed above will terminate in the final state s_3 if and only if the bit string has at least two bits and the second bit of the input string is a 1. If the bit string contains fewer than two bits or if the second bit is not a 1, the machine will terminate in the nonfinal state s_2. ■

Given a regular set, a Turing machine that always moves to the right can be built to recognize this set (as in Example 2). To build the Turing machine, first find a finite-state automaton that recognizes the set and then construct a Turing machine using the transition function of the finite-state machine, always moving to the right.

We will now show how to build a Turing machine that recognizes a nonregular set.

EXAMPLE 3 Find a Turing machine that recognizes the set $\{0^n 1^n \mid n \geq 1\}$.

Solution: To build such a machine, we will use an auxiliary tape symbol M as a marker. We have $V = \{0, 1\}$ and $I = \{0, 1, M\}$. We wish to recognize only strings in V^*. We will have one final state, s_6. The Turing machine successively replaces a 0 at the leftmost position of the string with an M and a 1 at the rightmost position of the string with an M, sweeping back and forth, terminating in a final state if and only if the string consists of a block of 0s followed by a block of the same number of 1s.

Although this is easy to describe and is easily carried out by a Turing machine, the machine we need to use is somewhat complicated. We use the marker M to keep track of the leftmost and rightmost symbols we have already examined. The five-tuples we use are: $(s_0, 0, s_1, M, R)$, $(s_1, 0, s_1, 0, R)$, $(s_1, 1, s_1, 1, R)$, (s_1, M, s_2, M, L), (s_1, B, s_2, B, L), $(s_2, 1, s_3, M, L)$, $(s_3, 1, s_3, 1, L)$, $(s_3, 0, s_4, 0, L)$, (s_3, M, s_5, M, R), $(s_4, 0, s_4, 0, L)$, (s_4, M, s_0, M, R), (s_5, M, s_6, M, R). For example, the string 000111 would successively become $M00111$, $M0011M$, $MM011M$, $MM01MM$, $MMM1MM$, $MMMMMM$ as the machine operates until it halts. (Note that this string is not changed by all steps of the operation of the Turing machine.)

We leave it to the reader (Exercise 13 at the end of this section) to explain the actions of this Turing machine and to explain why it recognizes the set $\{0^n 1^n \mid n \geq 1\}$. ■

It can be shown that a set can be recognized by a Turing machine if and only if it can be generated by a type 0 grammar, or in other words, if the set is generated by a phrase-structure grammar. The proof will not be presented here.

COMPUTING FUNCTIONS WITH TURING MACHINES

A Turing machine can be thought of as a computer that finds the values of a partial function. To see this, suppose that the Turing machine T, when given the string x as input, halts with the string y on its tape. We can then define $T(x) = y$. The domain of T is the set of strings for which T halts; $T(x)$ is undefined if T does not halt when given x as input. Thinking of a Turing machine as a machine that computes the values of a function on strings is useful, but how can we use Turing machines to compute functions defined on integers, on pairs of integers, on triples of integers, and so on?

To consider a Turing machine as a computer of functions from the set of k-tuples of nonnegative integers to the set of nonnegative integers (such functions are called **number-theoretic functions**), we need a way to represent k-tuples of integers on a tape. To do so, we use **unary representations** of integers. We represent the nonnegative integer n by a string of $n + 1$ 1s so that, for instance, 0 is represented by the string 1 and 5 is represented by the string 111111. To represent the k-tuple $(n_1, n_2, \ldots, n_k)$, we use a string of $n_1 + 1$ 1s, followed by an asterisk, followed by a string of $n_2 + 1$ 1s, followed by an asterisk, and so on, ending with a string of $n_k + 1$ 1s. For example, to represent the four-tuple $(2, 0, 1, 3)$ we use the string $111 * 1 * 11 * 1111$.

We can now consider a Turing machine T as computing a sequence of number-theoretic functions $T, T^2, \ldots, T^k, \ldots$. The function T^k is defined by the action of T on k-tuples of integers represented by unary representations of integers separated by asterisks.

EXAMPLE 4 Construct a Turing machine for adding two nonnegative integers.

Solution: We need to build a Turing machine T that computes the function $f(n_1, n_2) = n_1 + n_2$. The pair (n_1, n_2) is represented by a string of $n_1 + 1$ 1s followed by an asterisk followed by $n_2 + 1$ 1s. The machine T should take this as input and produce as output a tape with $n_1 + n_2 + 1$ ones. One way to do this is as follows. The machine starts at the leftmost 1 of the input string, and carries out steps to erase this 1, halting if $n_1 = 0$ so that there are no more 1s before the asterisk, replaces the asterisk with the leftmost remaining 1, and then halts. We can use the following five-tuples to do this: $(s_0, 1, s_1, B, R)$, $(s_1, *, s_3, B, R)$, $(s_1, 1, s_2, B, R)$, $(s_2, 1, s_2, 1, R)$, $(s_2, *, s_3, 1, R)$. ■

Unfortunately, constructing Turing machines to compute relatively simple functions can be extremely demanding. For example, one Turing machine for multiplying two nonnegative integers found in many books has 31 five-tuples and 11 states. If it is challenging to construct Turing machines to compute even relatively simple functions, what hope do we have of building Turing machines for more complicated functions? One way to simplify this problem is to use a multitape Turing machine that uses more than one tape simultaneously and to build up multitape Turing machines for the composition of functions. It can be shown that for any multitape Turing machine there is a one-tape Turing machine that can do the same thing.

A function that can be computed by a Turing machine is called *computable*. It is fairly straightforward to show that there are number-theoretic functions which are not computable. However, it is not so easy to actually produce such a function. The *busy beaver function* defined in the preamble to Exercise 23 at the end of this section is an example of a noncomputable function. One way to show that the busy beaver function is not computable is to show that it grows faster than any computable function. (See Exercise 24.)

DIFFERENT TYPES OF TURING MACHINES

There are many variations on the definition of a Turing machine. We can expand the capabilities of a Turing machine in a wide variety of ways. For example, we can allow a

Turing machine to move right, left, or not at all at each step. We can allow a Turing machine to operate on multiple tapes, using $(2+3n)$-tuples to describe the Turing machine when n tapes are used. We can allow the tape to be two-dimensional, where at each step we move up, down, right, or left, not just right or left as we do on a one-dimensional tape. We can allow multiple tape heads that read different cells simultaneously. Furthermore, we can allow a Turing machine to be nondeterministic, by allowing a (state, tape symbol) pair to possibly appear as the first elements in more than one five-tuple of the Turing machine. We can also reduce the capabilities of a Turing machine in different ways. For example, we can restrict the tape to be infinite in only one dimension or we can restrict the tape alphabet to have only two symbols. All these variations of Turing machines have been studied in detail.

The crucial point is that no matter which of these variations we use, or even which combination of variations we use, we never increase or decrease the power of the machine. Anything that one of these variations can do can be done by the Turing machine defined in this section, and vice versa. The reason that these variations are useful is that sometimes they make doing some particular job much easier than if the Turing machine defined in Definition 1 were used. They never extend the capability of the machine.

THE CHURCH–TURING THESIS

Turing machines are relatively simple. They can have only finitely many states and they can read and write only one symbol at a time on a one-dimensional tape. But it turns out that Turing machines are extremely powerful. We have seen that Turing machines can be built to add numbers and to multiply numbers. Although it may be difficult to actually construct a Turing machine to compute a particular function that can be computed with an algorithm, such a Turing machine can always be found. This was the original goal of Turing when he invented his machines.

Furthermore, there is a tremendous amount of evidence for the **Church–Turing thesis,** which states that given any problem which can be solved with an effective algorithm, there is a Turing machine that can solve this problem. The reason this is called a *thesis* rather than a theorem is that the concept of solvability by an effective algorithm is informal and imprecise, as opposed to the notion of solvability by a Turing machine, which is formal and precise. Certainly, though, any problem that can be solved using a computer with a program written in any language, perhaps using an unlimited amount of memory, should be considered effectively solvable.

Many different formal theories have been developed to capture the notion of effective computability. These include Turing's theory and Church's lambda-calculus, as well as theories proposed by Kleene and by Post. These theories seem quite different on the surface. The surprising thing is that they can be shown to be equivalent by demonstrating that they define exactly the same class of functions. With this evidence, it seems that Turing's original ideas, formulated before the invention of modern computers, describe the ultimate capabilities of these machines.

Alonzo Church (1903–1995). Alonzo Church was born in Washington, D.C. He studied at Göttingen under Hilbert and later in Amsterdam. He was a member of the faculty at Princeton University from 1927 until 1967 when he moved to UCLA. Church was one of the founding members of the Association for Symbolic Logic. He made many substantial contributions to the theory of computability, including his solution to the decision problem, his invention of the lambda-calculus, and, of course, his statement of what is now known as the Church–Turing thesis. Among Church's students were Stephen Kleene and Alan Turing. He published articles past his 90th birthday.

Exercises

1. Let T be the Turing machine defined by the five-tuples: $(s_0, 0, s_1, 1, R)$, $(s_0, 1, s_1, 0, R)$, $(s_0, B, s_1, 0, R)$, $(s_1, 0, s_2, 1, L)$, $(s_1, 1, s_1, 0, R)$, $(s_1, B, s_2, 0, L)$. For each of the following initial tapes, determine the final tape when T halts, assuming that T begins in initial position.

a)	⋯	B	B	0	0	1	1	B	B	⋯
b)	⋯	B	B	1	0	1	B	B	B	⋯
c)	⋯	B	B	1	1	B	0	1	B	⋯
d)	⋯	B	B	B	B	B	B	B	B	⋯

2. Let T be the Turing machine defined by the five-tuples: $(s_0, 0, s_1, 0, R)$, $(s_0, 1, s_1, 0, L)$, $(s_0, B, s_1, 1, R)$, $(s_1, 0, s_2, 1, R)$, $(s_1, 1, s_1, 1, R)$, $(s_1, B, s_2, 0, R)$, $(s_2, B, s_3, 0, R)$. For each of the following initial tapes, determine the final tape when T halts, assuming that T begins in initial position.

a)	⋯	B	B	0	1	0	1	B	B	⋯
b)	⋯	B	B	1	1	1	B	B	B	⋯
c)	⋯	B	B	0	0	B	0	0	B	⋯
d)	⋯	B	B	B	B	B	B	B	B	⋯

3. What does the Turing machine described by the five-tuples $(s_0, 0, s_0, 0, R)$, $(s_0, 1, s_1, 0, R)$, (s_0, B, s_2, B, R), $(s_1, 0, s_1, 0, R)$, $(s_1, 1, s_0, 1, R)$, and (s_1, B, s_2, B, R) do when given a bit string as input?

4. What does the Turing machine described by the five-tuples $(s_0, 0, s_1, B, R)$, $(s_0, 1, s_1, 1, R)$, $(s_1, 0, s_1, 0, R)$, $(s_1, 1, s_2, 1, R)$, $(s_2, 0, s_1, 0, R)$, $(s_2, 1, s_3, 0, L)$, $(s_3, 0, s_4, 0, R)$, and $(s_3, 1, s_4, 0, R)$ do when given a bit sting as input?

5. Construct a Turing machine with tape symbols 0, 1, and B that replaces the first 0 with a 1 and does not change any of the other symbols on the tape.

6. Construct a Turing machine with tape symbols 0, 1, and B that, given a bit string as input, replaces all 0s on the tape with 1s and does not change any of the 1s on the tape.

7. Construct a Turing machine with tape symbols 0, 1, and B that, given a bit string as input, replaces all but the leftmost 1 on the tape with 0s and does not change any of the other symbols on the tape.

8. Construct a Turing machine with tape symbols 0, 1, and B that, given a bit string as input, replaces the first two consecutive 1s on the tape with 0s and does not change any of the other symbols on the tape.

9. Construct a Turing machine that recognizes the set of all bit strings which end with a 0.

10. Construct a Turing machine that recognizes the set of all bit strings which contain at least two 1s.

11. Construct a Turing machine that recognizes the set of all bits strings that contain an even number of 1s.

12. Show at each step the contents of the tape of the Turing machine in Example 3 starting with each of the following strings:

a) 0011 **b)** 00011 **c)** 101100 **d)** 000111

13. Explain why the Turing machine in Example 3 recognizes a bit string if and only if this string is of the form 0^n1^n for some positive integer n.

***14.** Construct a Turing machine that recognizes the set $\{0^{2n}1^n \mid n \geq 0\}$.

***15.** Construct a Turing machine that recognizes the set $\{0^n1^n2^n \mid n \geq 0\}$.

16. Construct a Turing machine that computes the function $f(n) = n + 2$ for all nonnegative integers n.

17. Construct a Turing machine that computes the function $f(n) = n - 3$ if $n \geq 3$ and $f(n) = 0$ for $n = 0, 1, 2$ for all nonnegative integers n.

18. Construct a Turing machine that computes the function $f(n) = n$ **mod** 3.

19. Construct a Turing machine that computes the function $f(n) = 3$ if $n \geq 5$ and $f(n) = 0$ if $n = 0, 1, 2, 3,$ or 4.

20. Construct a Turing machine that computes the function $f(n_1, n_2) = n_2 + 2$ for all pairs of nonnegative integers n_1 and n_2.

***21.** Construct a Turing machine that computes the function $f(n_1, n_2) = \min(n_1, n_2)$ for all nonnegative integers n_1 and n_2.

22. Construct a Turing machine that computes the function $f(n_1, n_2) = n_1 + n_2 + 1$ for all nonnegative integers n_1 and n_2.

web Let $B(n)$ be the maximum number of 1s that a Turing machine with n states with the alphabet $\{1, B\}$ may print on a tape which is initially blank. The problem of determining $B(n)$ for particular values of n is known as the **busy beaver problem.** This problem was first studied by Tibor Rado in 1962. Currently it is known that $B(2) = 4$, $B(3) = 6$, and $B(4) = 13$, but $B(n)$ is not known for $n \geq 5$.

***23.** Show that $B(2)$ is at least 4 by finding a Turing machine with two states and alphabet $\{1, B\}$ that halts with four consecutive 1s on the tape.

****24.** Show that the function $B(n)$ cannot be computed by any Turing machine. [*Hint:* Assume that there is a Turing machine which computes $B(n)$ in binary. Build a Turing machine T that, starting with a blank tape, writes n down in binary, computes $B(n)$ in binary, and converts $B(n)$ from binary to unary. Show that for sufficiently large n, the number of states of T is less than $B(n)$, leading to a contradiction.]

Key Terms and Results

TERMS

alphabet (or vocabulary): a set that contains elements used to form strings

language: a subset of the set of all strings over an alphabet

phrase-structure grammar (V, T, S, P): a description of a language containing an alphabet V, a set of terminal symbols T, a start symbol S, and a set of productions P

the production $w \to w_1$: w can be replaced by w_1 whenever it occurs in a string in the language

$w_1 \Rightarrow w_2$ (w_2 is directly derivable from w_1): w_2 can be obtained from w_1 using a production to replace a string in w_1 with another string

$w_1 \overset{*}{\Rightarrow} w_2$ (w_2 is derivable from w_1): w_2 can be obtained from w_1 using a sequence of productions to replace strings by other strings

type 0 grammar: any phrase-structure grammar

type 1 grammar: a phrase-structure grammar in which every production is of the form $w_1 \to w_2$, where $l(w_1) \le l(w_2)$ or $w_2 = \lambda$

type 2, or context-free, grammar: a phrase-structure grammar in which every production is of the form $A \to w_1$, where A is a nonterminal symbol

type 3, or regular, grammar: a phrase-structure grammar where every production is of the form $A \to aB$, $A \to a$, or $S \to \lambda$, where A and B are nonterminal symbols, S is the start symbol, and a is a terminal symbol

derivation (or parse) tree: an ordered rooted tree where the root represents the starting symbol of a type 2 grammar, internal vertices represent nonterminals, leaves represent terminals, and the children of a vertex are the symbols on a right side of a production, in order from left to right, where the symbol represented by the parent is on the left-hand side

Backus–Naur form: a description of a context-free grammar in which all productions having the same nonterminal as their left-hand side are combined with the different right-hand sides of these productions, each separated by a bar, with nonterminal symbols enclosed in angular brackets and the symbol $\to$ replaced by ::=

finite-state machine (S, I, O, f, g, s_0) (or a Mealy machine): a six-tuple containing a set S of states, an input alphabet I, an output alphabet O, a transition function f that assigns a next state to every pair of a state and an input, an output function g that assigns an output to every pair of a state and an input, and a starting state s_0

AB (concatenation of A and B): the set of all strings formed by concatenating a string in A and a string in B in that order

A^* (Kleene closure of A): the set of all strings made up by concatenating arbitrarily many strings from A

deterministic finite-state automaton (S, I, f, s_0, F): a five-tuple containing a set S of states, an input alphabet I, a transition function f that assigns a next state to every pair of a state and an input, a starting state s_0, and a set of final states F

nondeterministic finite-state automaton (S, I, f, s_0, F): a five-tuple containing a set S of states, an input alphabet I, a transition function f that assigns a set of possible next states to every pair of a state and an input, a starting state s_0, and a set of final states F

language recognized by an automaton: the set of input strings that take the start state to a final state of the automaton

regular expression: an expression defined recursively by specifying that $\emptyset$, λ, and x, for all x in the input alphabet, are regular expressions, and that (AB), (A ∪ B), and (A)* are regular expressions when A and B are regular expressions

regular set: a set defined by a regular expression (see page 656)

Turing machine $T = (S, I, f, s_0)$: a four-tuple consisting of a finite set S of states, an alphabet I containing the blank symbol B, a partial function f from $S \times I$ to $S \times I \times \{R, L\}$, and a starting state s_0

RESULTS

For any nondeterministic finite-state automaton there is a deterministic finite-state automaton that recognizes the same set.

Kleene's theorem: A set is regular if and only if there is a finite-state automaton that recognizes it.

A set is regular if and only if it is generated by a regular grammar.

Review Questions

1. **a)** Define a phrase-structure grammar.
   b) What does it mean for a string to be derivable from a string w by a phrase-structure grammar G?
2. **a)** What is the language generated by a phrase-structure grammar G?
   b) What is the language generated by the grammar G with vocabulary $\{S, 0, 1\}$, set of terminals $T = \{0, 1\}$, starting symbol S, and productions $S \to 000S$, $S \to 1$?
   c) Give a phrase-structure grammar that generates the set $\{01^n \mid n = 0, 1, 2, \ldots\}$.
3. **a)** Define a type 1 grammar.
   b) Give an example of a grammar that is not a type 1 grammar.
   c) Define a type 2 grammar.
   d) Give an example of a grammar that is not a type 2 grammar but is a type 1 grammar.
   e) Define a type 3, or context-free, grammar.
   f) Give an example of a grammar that is not a type 3 grammar but is a type 2 grammar.
4. **a)** Define a regular grammar.
   b) Define a regular language.
   c) Give an example of a grammar that is not regular but is a type 3 grammar.
   d) Show that the set $\{0^m1^n \mid m, n = 0, 1, 2, \ldots\}$ is a regular language.
5. **a)** What is Backus–Naur form?
   b) Give an example of the Backus–Naur form of the grammar for a subset of English of your choice.
6. **a)** What is a finite-state machine?
   b) Show how a vending machine that accepts only quarters and dispenses a soft drink after 75 cents has been deposited can be modeled using a finite-state machine.
7. **a)** What is the Kleene closure of a set of strings?
   b) Find the Kleene closure of the set $\{11, 0\}$.
8. **a)** Define a finite-state automaton.
   b) What does it mean for a string to be recognized by a finite-state automaton?
9. **a)** Define a nondeterministic finite-state automaton.
   b) Show that given a nondeterministic finite-state automaton, there is a deterministic finite-state automaton that recognizes the same language.
10. **a)** Define the set of regular expressions over a set I.
    b) Explain how regular expressions are used to represent regular sets.
11. State Kleene's Theorem.
12. Show that a set is generated by a regular grammar if and only if it is a regular set.
13. Give an example of a set not recognized by a finite-state automaton. Show that no finite-state automaton recognizes it.
14. Define a Turing machine.
15. Describe how Turing machines are used to recognize sets.
16. Describe how Turing machines are used to compute number theoretic functions.

Supplementary Exercises

***1.** Find a phrase-structure grammar that generates each of the following languages.
   a) the set of bit strings of the form $0^{2n}1^{3n}$ where n is a nonnegative integer
   b) the set of bit strings with twice as many 0s as 1s
   c) the set of bit strings of the form w^2 where w is a bit string

***2.** Find a phrase-structure grammar that generates the set $\{0^{2^n} \mid n \geq 0\}$.

For Exercises 3 and 4, let $G = (V, T, S, P)$ be the context-free grammar with $V = \{(,), S, A, B\}$, $T = \{(,)\}$, starting symbol S, and productions $S \to A$, $A \to AB$, $A \to B$, $B \to (A)$, and $B \to ( )$, $S \to \lambda$.

3. Construct the derivation trees of the following.
   a) (( )) **b)** ( )(( )) **c)** ((( )( )))

***4.** Show that $L(G)$ is the set of all well-formed strings of parentheses, defined in Chapter 3.

A context-free grammar is **ambiguous** if there is a word in $L(G)$ with two derivations that produce different derivation trees, considered as ordered, rooted trees.

5. Show that the grammar $G = (V, T, S, P)$ with $V = \{0, S\}$, $T = \{0\}$, starting state S, and productions $S \to 0S$, $S \to S0$, and $S \to 0$ is ambiguous by constructing two different derivation trees for 0^3.

6. Show that the grammar $G = (V, T, S, P)$ with $V = \{0, S\}$, $T = \{0\}$, starting state S, and productions $S \to 0S$ and $S \to 0$ is unambiguous.

7. Suppose that A and B are finite subsets of V^*, where V is an alphabet. Is it necessarily true that $|AB| = |BA|$?

8. Prove or disprove each of the following statements for subsets A, B, and C of V^*, where V is an alphabet.

a) $A(B \cup C) = AB \cup AC$

b) $A(B \cap C) = AB \cap AC$

c) $(AB)C = A(BC)$

d) $(A \cup B)^* = A^* \cup B^*$

9. Suppose that A and B are subsets of V^*, where V is an alphabet. Does it follow that $A \subseteq B$ if $A^* \subseteq B^*$?

10. What set of strings with symbols in the set $\{0, 1, 2\}$ is represented by the regular expression $\mathbf{(2^*)(0 \cup (12^*))^*}$?

The **star height** $h(\mathbf{E})$ of a regular expression over the set I is defined recursively by

$$h(\emptyset) = 0;$$
$$h(\mathbf{x}) = 0 \text{ if } \mathbf{x} \in I;$$
$$h((\mathbf{E}_1 \cup \mathbf{E}_2)) = h((\mathbf{E}_1\mathbf{E}_2)) = \max(h(\mathbf{E}_1), h(\mathbf{E}_2))$$
$$\text{if } \mathbf{E}_1 \text{ and } \mathbf{E}_2 \text{ are regular expressions;}$$
$$h(\mathbf{E}^*) = h(\mathbf{E}) + 1 \text{ if } \mathbf{E} \text{ is a regular expression.}$$

11. Find the star height of each of the following regular expressions.

a) $\mathbf{0^*1}$ **b)** $\mathbf{0^*1^*}$

c) $\mathbf{(0^*01)^*}$ **d)** $\mathbf{((0^*1)^*)^*}$

e) $\mathbf{(010^*)(1^*01^*)^*((01)^*(10)^*)^*}$

f) $\mathbf{((((0^*)1)^*0)^*)1)^*}$

***12.** For each of the following regular expressions find a regular expression that represents the same language with minimum star height.

a) $\mathbf{(0^*1^*)^*}$

b) $\mathbf{(0(01^*0)^*)^*}$

c) $\mathbf{(0^* \cup (01)^* \cup 1^*)^*}$

13. Construct a finite-state machine with output that produces an output of 1 if the bit string read so far as input contains four or more 1s. Then construct a deterministic finite-state automaton that recognizes this set.

14. Construct a finite-state machine with output that produces an output of 1 if the bit string read so far as input contains four or more consecutive 1s. Then construct a deterministic finite-state automaton that recognizes this set.

15. Construct a finite-state machine with output that produces an output of 1 if the bit string read so far as input ends with four or more consecutive 1s. Then construct a deterministic finite-state automaton that recognizes this set.

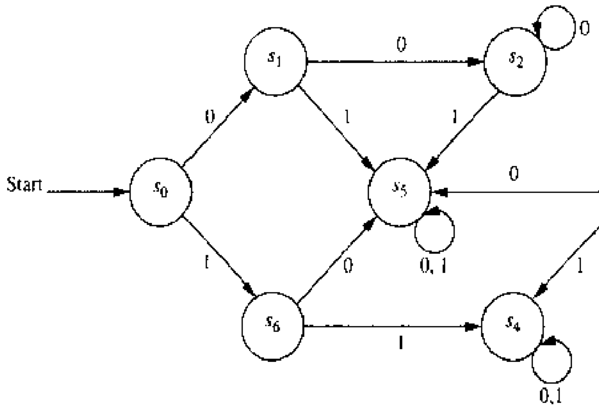
16. A state s' in a finite-state machine is said to be **reachable** from state s if there is an input string x such that $f(s, x) = s'$. A state s is called **transient** if there is no nonempty input string x with $f(s, x) = s$. A state s is called a **sink** if $f(s, x) = s$ for all input strings x. Answer questions **a–d** about the finite-state machine with the state diagram illustrated below.

a) Which states are reachable from s_0?

b) Which states are reachable from s_2?

c) Which states are transient?

d) Which states are sinks?



***17.** Suppose that S, I, and O are finite sets such that $|S| = n$, $|I| = k$, and $|O| = m$.

a) How many different finite-state machines (Mealy machines) $M = (S, I, O, f, g, s_0)$ can be constructed, where the starting state s_0 can be arbitrarily chosen?

b) How many different Moore machines $M = (S, I, O, f, g, s_0)$ can be constructed, where the starting state s_0 can be arbitrarily chosen?

***18.** Suppose that S and I are finite sets such that $|S| = n$ and $|I| = k$. How many different finite-state automata $M = (S, I, f, s_0, F)$ are there where the starting state s_0 and the subset F of S consisting of final states can be chosen arbitrarily

a) if the automata are deterministic?

b) if the automata may be nondeterministic? (*Note:* This includes deterministic automata.)

19. Construct a deterministic finite-state automaton that is equivalent to the nondeterministic automaton with the state diagram at the bottom of this column.

20. What is the language recognized by the automaton in Exercise 19?

21. Construct finite-state automata that recognize the following sets.

a) $0^*(10)^*$ **b)** $(01 \cup 111)^*10^*(0 \cup 1)$

c) $(001 \cup (11)^*)^*$

***22.** Find regular expressions that represent the set of all strings of 0s and 1s

a) made up of blocks of even numbers of 1s interspersed with odd numbers of 0s.

b) with at least two consecutive 0s or three consecutive 1s.

c) with no three consecutive 0s or two consecutive 1s.

***23.** Show that if A is a regular set, then so is $\overline{A}$.

***24.** Show that if A and B are regular sets, then so is $A \cap B$.

***25.** Find finite-state automata that recognize the following sets of strings of 0s and 1s.

a) the set of all strings that start with no more than three consecutive 0s and contain at least two consecutive 1s

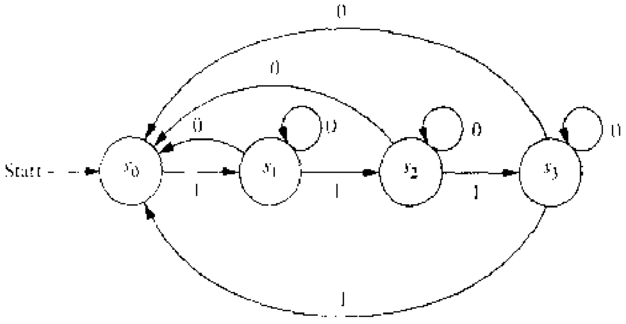
b) the set of all strings with an even number of symbols that do not contain the pattern 101

c) the set of all strings with at least three blocks of two or more 1s and at least two 0s

***26.** Show that $\{0^{2^n} \mid n \in \mathbf{Z}\}$ is not regular. You may use the pumping lemma given in Exercise 16 of Section 10.4.

***27.** Show that $\{1^p \mid p \text{ is prime}\}$ is not regular. You may use the pumping lemma given in Exercise 16 of Section 10.4.

***28.** There is a result for context-free languages analogous to the pumping lemma for regular sets. Suppose that $L(G)$ is the language recognized by a context-free language G. This result states that there is a constant N such that if z is a word in $L(G)$ with $l(w) \geq N$, then z can be written as $uvwxy$ where $l(vwx) \leq N$, $l(vx) \geq 1$, and uv^iwx^iy belongs to $L(G)$ for $i = 0, 1, 2, 3, \ldots$. Use this result to show that there is no context-free grammar G with $L(G) = \{0^n1^n2^n \mid n = 0, 1, 2, \ldots\}$.



Computer Projects

WRITE PROGRAMS WITH THE FOLLOWING INPUT AND OUTPUT.

1. Given the productions in a phrase-structure grammar, determine which type of grammar this is in the Chomsky classification scheme.

***2.** Given the productions of a context-free grammar and a string, produce a derivation tree for this string if it is in the language generated by this grammar.

3. Given the state table of a Moore machine and an input string, produce the output string generated by the machine.

4. Given the state table of a Mealy machine and an input string, produce the output string generated by the machine.

5. Given the state table of a deterministic finite-state automaton and a string, decide whether this string is recognized by the automaton.

6. Given the state table of a nondeterministic finite-state automaton and a string, decide whether this string is recognized by the automaton.

***7.** Given the state table of a nondeterministic finite-state automaton, construct the state table of a deterministic finite-state automaton that recognizes the same language.

****8.** Given a regular expression, construct a nondeterministic finite-state automaton that recognizes the set that this expression represents.

9. Given a regular grammar, construct a finite-state automaton that recognizes the language generated by this grammar.

10. Given a finite-state automaton, construct a regular grammar that generates the language recognized by this automaton.

***11.** Given a Turing machine, find the output string produced by a given input string.

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. Solve the busy beaver problem for two states by testing all possible Turing machines with two states and alphabet $\{1, B\}$.

***2.** Solve the busy beaver problem for three states by testing all possible Turing machines with three states and alphabet $\{1, B\}$.

****3.** Find a busy beaver machine with four states by testing all possible Turing machines with four states and alphabet $\{1, B\}$.

****4.** Make as much progress as you can toward finding a busy beaver machine with five states.

****5.** Make as much progress as you can toward finding a busy beaver machine with six states.

Writing Projects

RESPOND TO THE FOLLOWING QUESTIONS WITH ESSAYS USING OUTSIDE SOURCES.

1. Describe how the growth of certain types of plants can be modeled using a Lidenmeyer system. Such a system uses a grammar with productions modeling the different ways plants can grow.

2. Describe the Backus–Naur form rules used to specify the syntax of different programming languages, including Java, LISP, ADA, and the database language SQL.

3. Explain how finite-state machines are used in the study of network protocols.

4. Explain the concept of minimizing finite-state automata. Give an algorithm that carries out this minimization.

5. Give the definition of cellular automaton. Explain their applications. Use the Game of Life as an example.

6. Define a pushdown automaton. Explain how pushdown automata are used to recognize sets. Which sets are recognized by pushdown automata? Provide an outline of a proof justifying your answer.

7. Define a linear-bounded automaton. Explain how linear-bounded automata are used to recognize sets. Which sets are recognized by linear-bounded automata? Provide an outline of a proof justifying your answer.

8. Look up Turing's original definition of what we now call a Turing machine. What was his motivation for defining these machines?

9. Describe the concept of the universal Turing machine. Explain how such a machine can be built.

10. Explain the kinds of applications in which nondeterministic Turing machines are used instead of deterministic Turing machines.

11. Show that a Turing machine can simulate any action of a nondeterministic Turing machine.

12. Show that a set is recognized by a Turing machine if and only if it is generated by a phrase-structure grammar.

13. Describe the basic concepts of the lambda-calculus and explain how it is used to study computability of functions.

14. Show that a Turing machine as defined in this chapter can do anything a Turing machine with n tapes can do.

15. Show that a Turing machine with a tape infinite in one direction can do anything a Turing machine with a tape infinite in both directions can do.

Computations and Explorations

USE A COMPUTATIONAL PROGRAM OR PROGRAMS YOU HAVE WRITTEN TO DO THE FOLLOWING EXERCISES.

1. Solve the busy beaver problem for two states by testing all possible Turing machines with two states and alphabet $\{1, B\}$.

***2.** Solve the busy beaver problem for three states by testing all possible Turing machines with three states and alphabet $\{1, B\}$.

****3.** Find a busy beaver machine with four states by testing all possible Turing machines with four states and alphabet $\{1, B\}$.

****4.** Make as much progress as you can toward finding a busy beaver machine with five states.

****5.** Make as much progress as you can toward finding a busy beaver machine with six states.

Writing Projects

RESPOND TO THE FOLLOWING QUESTIONS WITH ESSAYS USING OUTSIDE SOURCES.

1. Describe how the growth of certain types of plants can be modeled using a Lidenmeyer system. Such a system uses a grammar with productions modeling the different ways plants can grow.

2. Describe the Backus–Naur form rules used to specify the syntax of different programming languages, including Java, LISP, ADA, and the database language SQL.

3. Explain how finite-state machines are used in the study of network protocols.

4. Explain the concept of minimizing finite-state automata. Give an algorithm that carries out this minimization.

5. Give the definition of cellular automaton. Explain their applications. Use the Game of Life as an example.

6. Define a pushdown automaton. Explain how pushdown automata are used to recognize sets. Which sets are recognized by pushdown automata? Provide an outline of a proof justifying your answer.

7. Define a linear-bounded automaton. Explain how linear-bounded automata are used to recognize sets. Which sets are recognized by linear-bounded automata? Provide an outline of a proof justifying your answer.

8. Look up Turing's original definition of what we now call a Turing machine. What was his motivation for defining these machines?

9. Describe the concept of the universal Turing machine. Explain how such a machine can be built.

10. Explain the kinds of applications in which nondeterministic Turing machines are used instead of deterministic Turing machines.

11. Show that a Turing machine can simulate any action of a nondeterministic Turing machine.

12. Show that a set is recognized by a Turing machine if and only if it is generated by a phrase-structure grammar.

13. Describe the basic concepts of the lambda-calculus and explain how it is used to study computability of functions.

14. Show that a Turing machine as defined in this chapter can do anything a Turing machine with n tapes can do.

15. Show that a Turing machine with a tape infinite in one direction can do anything a Turing machine with a tape infinite in both directions can do.